R E V I S T A   P A R A   U S U A R I O S   D E

Drean Commodore

Nº 5 ₳ 2.00

REP. ARGENTINA



√ Compútelo a su favor

En Argecint la Drean Commodore trae una tecla más: el respaldo.



El respaldo, el service y el asesoramiento que puede ofrecer una empresa con 15 años, dedicados a la comercialización de todo lo que tenga que ver con el mundo de la computación. Argecint, El Súper Todo de Computación. Que pone a su alcance la Drean Commodore, con la línea más completa de accesorios, programas, periféricos... y con la financiación más cómoda de plaza y donde siempre encontrará un asesoramiento de expertos y una atención de amigos.

ARGEcint

el Súper Todo de Computación

COMPUTADORES
PERIFERICOS
MAGNETICOS
MUEBLES
CINTAS
CASSETTES
ACCESORIOS
SUMINISTROS
FORMULARIOS
LAB. TECNICO
CURSOS DE COMPUTACION

Drean Commodore y Argecint, la combinación perfecta.

PLANES UNICOS DE FINANCIACION - TARJETAS DE CREDITO

Casa Matriz: VENTURA BOSCH 7065 - Tel.: 641-0527/4892/3051 TELEX 17312 (ERSA) C.C. 8 Suc. 8 (1408) Cap. Fed.
Casa Central: AV. DE MAYO 1402 - Tel.: 37-4631 - Cap. Fed.
Agencia Trust: CARLOS PELLEGRINI Y CORRIENTES - Tel.: 35-5018/5029/0344 - Cap. Fed.
Agencia Belgrano: COMPUMARKET - AV. CABILDO 2869/71 - Tel.: 785-5241/4689 - Cap. Fed.
Agencia Flores: TRUST JOYERO - AV. RIVADAVIA 6687 - Tel.: 634-4639 - Cap. Fed.
Agencia Avellaneda: HIJOS DE G. ROSSI - AV. MITRE 660 - Tel. 201-5658 - Bs. As.
Sucursal Liniers: AV. RIVADAVIA 11332 (1408) - Tel.: 641-3088 - Cap. Fed.
Agencia Litoral: PEATONAL SAN MARTIN 2455 - Loc. 36 (3000) - Tel.: 25459 - STA. FE
Agencia Barrio Norte: AV. SANTA FE 2646 - Tel.: 84-8870 - Cap. Fed.
Agencia Computer Beach: AV. J.C. CHIOZZA 2872 (6561) - Tel.: 291 - SAN BERNARDO - BS. AS.

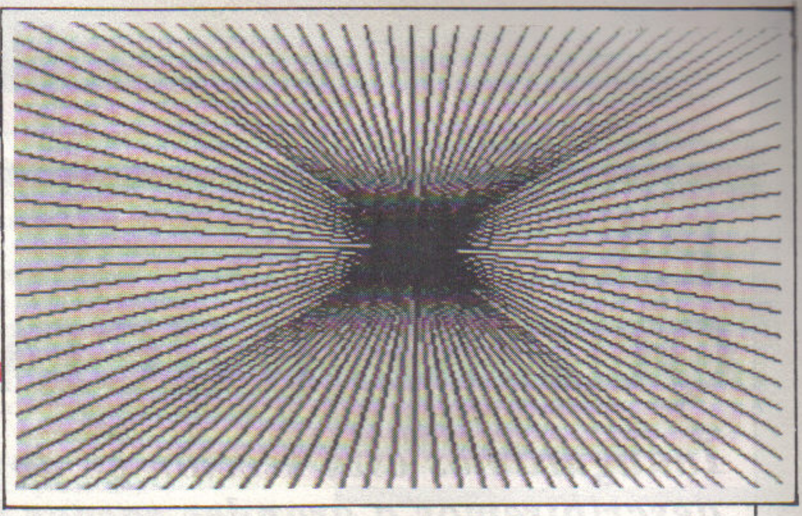
SUMARIO

NOTAS TECNICAS

PROGRAMAS

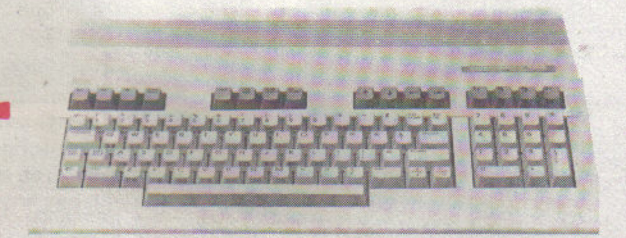
REVISION DE SOFTWARE

SECCIONES FIJAS



Explicamos cómo la C-16 permite la creación de las más diversas figuras.

Presentamos uno de los periféricos recientemente lanzados para la 128.





Continuamos comentando y calificando los programas que Peek presentó en el mercado argentino

Drean Commodore AÑO 1 Nº 5 ABRIL DE 1986

Director General
Ernesto del Castillo
Director Editorial
Cristian Pusso
Director Periodístico
Fernando Flores

Director Financiero
Javier Campos Malbrán
Arte y Diagramación
Fernando Amengual
Coordinador
Ariel Testori

Redacción
Cristian Parodi
Departamento de Avisos
Oscar Devoto
Departamento de Publicidad
Guillermo Gonzalez Aldalur

Drean Commodore es una Revista mensual editada por Editorial PROEDI S.A. Parana 720. 5º Piso. (1017) Buenos Aires. Tel.: 46-2886 y 49-7130. Reg. Nac. de la Prop. Intelectual E.T., M. Registrada

Precio de este ejemplar: ₳ 2.
Impresion: Calcotam. Fotocromo tapa: Columbia. Fotocomposicion: Van Waveren.
Los ejemplares atrasados se venderán al precio del último número en circulacion.

Distribuidor en Capital: Martino, Juan de Garay 358, P.B. Capital. Distribuidor interior: DGP: Hipolito Yrigoyen 1450, Capital Federal. T.E. 38-9266/9800

PARA LOS QUE SE INICIAN

Inauguramos esta sección en donde describiremos a grandes rasgos los contenidos de las notas técnicas publicadas en cada número.



A partir de este número comenzaremos a describir, en esta sección, cada una de las notas técnicas que aparezcan.
Queremos de esta manera disminuir un poco la distancia que existe entre alguna de aquellas notas y el lector que se inicia en la computación.
Comenzaremos describiendo la nota referente al mapa de memoria de la Drean Commodore 64. Toda computadora está formada, básicamente, por el hardware (es decir el teclado, transistores, circuitos integrados, memorias, etc.) y por el software (los programas que administran los anteriores recursos). Cada uno de los programas necesitan, para funcionar correctamente, de una serie de datos. Estos están almacenados en una determinada sección de la memoria. Agreguemos que ésta se puede dividir en dos: la libre, en donde se nos permite escribir nuestros programas, y la que usa la computadora. Cada una de estas dos secciones están formadas por direcciones de memoria, las cuales contienen un solo dato. Es decir que en una serie de direcciones de memoria (una sección) se almacenan los datos necesarios para el funcionamiento de la computadora o para el funcionamiento de nuestro programa.
En "mapa de memoria" nosotros describimos la función de cada una de las direcciones que constituyen la sección que utiliza la C-64 para funcionar correctamente. Como verán más adelante, en la medida que profundicen sus conocimientos, conocer una determinada dirección de memoria permitirá mejorar los comandos del basic de la C-64. Además se requiere conocer un poco del lenguaje máquina del equipo.
En la nota de assembler se describe otra manera de programar la C-64: directamente en el lenguaje que él entiende. ¿Qué significa esto? Cada lenguaje de programación de los denominados de alto nivel (como el Basic), fue creado, entre otras cosas, para facilitarle la tarea al programador.
Por ejemplo una suma él la hará como LET A=C+D en basic, COMPUTE A = A + B en Cobol, etc. Realizar la misma operación en assembler equivale a ingresar en direcciones de memoria, direccionamientos, sumas en complemento a dos, y muchas otras cosas. Por esta razón, conviene más realizar un LET o un COMPUTE.
Pero a la hora de ejecutar el programa, estos símbolos deben ser traducidos al lenguaje que sí entiende la computadora y que es de bajo nivel. Este se llama, con justa razón, lenguaje máquina.
¿Cuál es la ventaja, pues, de trabajar con este lenguaje siendo tan poco práctico de programar? La respuesta es el tiempo de ejecución. La traducción que antes mencionamos demanda un tiempo. Al trabajar directamente en la "lengua" del computador, evitamos esta traducción logrando, así, mayor velocidad de ejecución. Cada juego, utilitario serio, etc., está escrito en lenguaje máquina.
En esta nota queremos realizar un curso introductorio al lenguaje assembler pasando primero por el lenguaje máquina. La diferencia que hay entre ellos es que el segundo es el que entiende directamente la computadora mientras que el primero son una serie de "palabras", denominadas mnemotécnicos, las cuales representan cada uno de los códigos que constituyen el lenguaje máquina. También se diseñó para facilitarle un poco las cosas al programador.
Insistimos: cada uno de los lenguajes (Assembler, Basic, Pascal, Cobol, etc.) se traducen siempre al lenguaje máquina.
En la nota "Las subrutinas del Drean Commodore 64" explicamos cómo usar los subprogramas que utiliza el sistema operativo. Este se encarga de administrar los recursos de la computadora como ser datos, periféricos, memoria y el procesador. Como antes mencionamos, la computadora necesita de determinados programas. En el caso de la C-64, ella necesita de los que comenzamos a describir en este número. En el próximo describiremos cómo acceder a ellos a través del basic.
Y seguimos con los gráficos en la C-16 comentando cómo hacer escenarios para nuestros primeros juegos y cómo realizar raras figuras utilizando los comandos que posee el equipo.

NUEVO LENGUAJE COMAL

La potencia de la programación estructurada del PASCAL, la facilidad del BASIC y los magníficos gráficos de la tortuga del LOGO se pueden obtener, en Estados Unidos, por solo u$s 7. Este precio corresponde al COMAL 0.14, lenguaje que contiene las anterior descripciones. Además, el lenguaje viene acompañado por una
Por una demostración de todas las operaciones que puede realizar este poderoso sistema. El diskett está desprotegido.

SUPERSHIPPER 64

SUPERSHIPPER 64 es un completo sistema de facturación y embar que de productos para la C-64. Para implementarlo se requieren de una o dos impresoras, unidad de disco simple o doble.
Algunas de las cosas que el sistema permiten son:
Menú de todas las operaciones 800 cuentas, 500 facturas y 200 productos por disco, expandible a 2200 productos con un drive adicional.
Fácil ingreso de datos, completo editor de "ventanas" para ingresos de facturas, cuentas y datos de productos.
Optimización del uso del disco y aumento de la velocidad de transferencia de datos.
Protejer nuestro sistema con dos niveles de acceso: Ejecutivo y operador.
Seleccionar formato de impresión de facturas, cuentas, etc.
Cuatro categorías de precio para cada uno de los productos.
Ordena e imprime los productos en orden alfabético.
Actualmente este programa utilitario no se encuentra disponible en nuestro mercado.

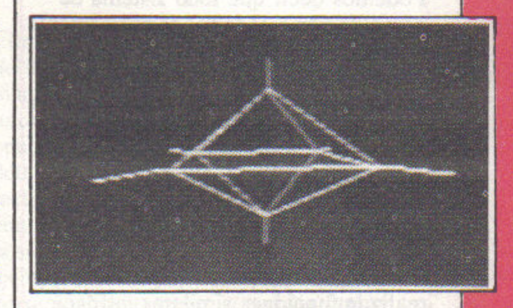
EXPANSION DE MEMORIA

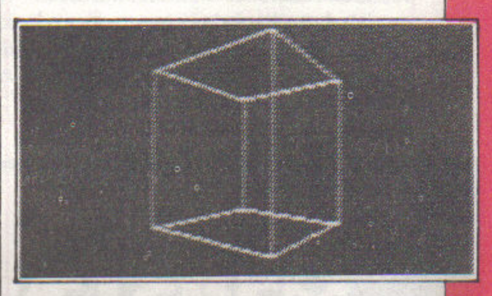
La firma americana Cardo Inc. lanzó al mercado norteamericano un cartridge que amplia la memoria RAM a mas de 60 kb. Además agrega un total de 60 nuevos comandos y funciones. Esta memoria no es de uso restringido: puede ser utilizada por variables, arrays (vectores, matrices, etc.), programas basic, etc. Los nuevos comandos son, entre otros, CATALOGO (imprime en pantalla el directorio del diskette actual), FIND, CHANGE, TRACE, DUMP, KEY. Las teclas de función son programables y pueden ser redefinidas. Por ejemplo podemos definir la tecla F2 para que ejecute el programa actual en memoria, F3 para que muestre el directorio del diskette, etc.

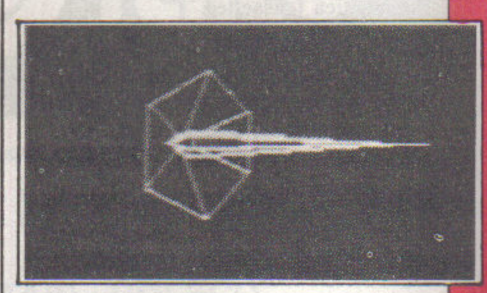
LA MUSICA Y LA C-64



El programa THE MUSIC SHOP es un potente utilitario que nos permite editar música en la C-64. Podemos almacenarla en diskette o cassette, realizar sonidos especiales y visualizar el pentagrama correspondiente con los tonos creados.







SISTEMA DE GRAFICOS EN TRES DIMENSIONES

Este sistema nos permite crear y manipular gráficos en tres dimensiones. Trabaja juntamente con un lápiz óptico. De esta manera podemos realizar gráficos (siempre en 3D) pudiendo variar su escala, rotarlos, cambiar los puntos de referencia de observación, etc. El usuario dispone de un menu el cual le permite seleccionar una función con solo tocarla con el lápiz óptico sobre la pantalla, logrando así un fácil manipuleo de todo el sistema.

UNIDAD DE CINTA

A través de estos accesorios podemos almacenar nuestros programas para más tarde utilizarlos. Son cien por cien compatibles con las C-64 y C-16.

Podemos decir que todo sistema de computación está formado básicamente por la unidad de control, la memoria principal y los periféricos de entrada-salida. Estos últimos son utilizados para ingresar datos desde el exterior (función que realiza, por ejemplo, el teclado) o para enviar datos hacia el exterior (aquí podríamos citar el monitor trabajando como un periférico de salida).

Existen, además, otros dispositivos que realizan funciones similares, es decir envían datos hacia la computadora o los reciben de ella. Además pueden almacenar tóda la información transmitida. De esta manera los programas que realizamos pueden ser "guardados" aquí para más tarde utilizarlos.

Hay dos medios magnéticos para realizar la operación antes descripta. Ellos son discos flexibles y cinta magnética (cassette).

El DATASSETTE es un periférico de entrada-salida, desarrollado para los equipos Drean Commodore 64 y 16, que nos permite almacenar nuestros programas y/o datos.

De acuerdo al equipo, la unidad de cinta puede ser modelo 1530 (compatible con la C-64) o la 1531 (compatible con la C-16). De todas formas las características que más adelante describiremos se adecúan a ambos modelos.

Similar a un grabador

Podríamos decir que el DATASSETTE es similar a un grabador comercial. Dispone de las teclas respectivas para rebobinar la cinta, grabar, avanzar, detener el movimiento, expulsar el cassette y, además, tiene un contador de cinta que nos permite determinar en qué lugar de ésta se encuentra un determinado programa.
Internamente cuenta con el cabezal para lectura/escritura, la cabeza de borrado y los rodillos transportadores de cinta.
Como la unidad es totalmente compatible con los equipos, no se necesita interface externa. Simplemente se la conecta al computador a través de un cable integrado, el cual, además de suministrar la energía necesaria para su funcionamiento, posibilita la comunicación entre éste y la computadora, la cual tiene una entrada especial para el DATASSETTE (para una mejor explicación miren el manual del usuario de la computadora).
Es aconsejable que la conexión se realice con la computadora apagada. Sobre la unidad se encuentra un pequeño indicador rojo de lectura/escritura.

Cómo se utiliza

Una vez que se realizaron todas las conexiones necesarias podemos comenzar a operar el DATASSETTE. Por ejemplo, podríamos escribir un pequeño programa para luego almacenarlo en el cassette. Para ello debemos utilizar el comando:
SAVE"nombre del programa"
y luego oprimir la tecla de return. En la pantalla aparecerá el mensaje "PRESS RECORD & PLAY ON TAPE". Aquí deberemos oprimir la tecla PLAY y RECORD del DATASSETTE simultáneamente. El indicador de escritura/lectura se encenderá indicando que el proceso de grabación se está realizando. Luego de unos segundos aparecerá el mensaje READY. De esta manera nuestro programa ya se encuentra almacenado en el cassette.
Si, en otro momento, queremos seguir trabajando sobre el programa o simplemente ejecutarlo, debemos realizar el proceso de lectura o carga. Esto se realiza efectuando:
LOAD"nombre del programa"
y, como antes, oprimiremos la tecla de return. En la pantalla ahora aparecerá el mensaje "PRESS PLAY ON TAPE". Aquí sólo debemos oprimir la tecla PLAY del DATASSETTE.
Seguidamente se emitirá "SEARCHING FOR nombre del programa", finalmente el mensaje READY indicando fin de carga.
Podemos verificar el proceso tipeando LIST. Debemos ver, si todo anduvo bien, el listado del programa. En caso de ocurrir algún error en la lectura de nuestro programa se nos informará con el correspondiente mensaje, como ser FILE NOT FOUND (nuestro programa no se encuentra en la cinta).
Es importante resaltar que tanto en lectura como en escritura de programas, hay que alejar el DATASSETTE del receptor de T.V. Esto se debe a que las ondas de video pueden interferir la información que va hacia o desde la computadora sin llegar a grabar o cargar los programas correctamente.
Como dijimos al principio, también se nos permite almacenar datos. Esto podemos hacerlo al trabajar con archivos, que por ser cinta magnética, son de organización secuencial. Pero el tratamiento de archivos será tema de próximas notas.

Mantenimiento periódico

Este es otro punto importante. No debemos olvidarnos de efectuar un mantenimiento periódico a la unidad. De esta manera aumentaremos su vida útil.
Cada pieza es vital para el correcto funcionamiento. La limpieza de los cabezales garantizará que los procesos de lectura y escritura estén libres de errores.
Muchos de los problemas de carga o de grabación de un determinado programa se deben a que la cabeza de lectura y/o grabación está sucia.
Otro de los problemas que comúnmente ocurren es la magnetización de los cabezales debido al acercamiento de un metal imantado. Para solucionarlo debemos recurrir a un cassette

DATASSETTE 1530-1531



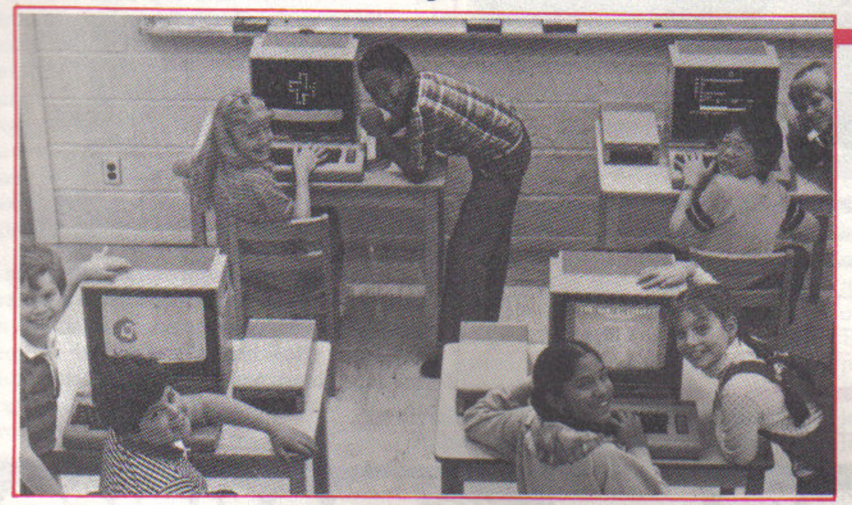
desmagnetizador, el cual puede ser adquirido en cualquier casa especializada.
Otro detalle es la elección del lugar donde vamos a guardar los cassettes y la unidad. Como primera medida, éste debe estar alejado de motores, televisores, metales, es decir de todos aquellos artefactos que puedan contener superficies imantadas.
Evitaremos, así, magnetizar los cabezales o perder la información almacenada en los cassettes.

Documentación

Como principal fuente de información del DATASSETTE podemos citar su manual, el cual contiene toda la información y datos necesarios para operarlo correctamente. Describe detalladamente cómo se almacenan y recuperan los programas, cómo trabajar con archivos y la solución a todos los posibles problemas. Además, varios ejemplos ilustran los procedimientos correctos. Juntamente con el manual se incluye la correspondiente garantía de la unidad y las direcciones de los services autorizados por DREAN.

LA COMPUTADORA COMO AUXILIAR DEL DOCENTE

Realizamos un análisis de la educación y la informática. Descubrimos, también, las posibilidades de los equipos Drean Commodore frente a la educación informatizada.



Actualmente se trabaja sobre la idea de la educación asistida por la computadora, sin descuidar la importantísima misión y función del educador. Claro que, a partir de aquí, éste tendrá un nuevo colaborador, quien lo ayudará en la enseñanza de cada materia.

Analizando la escuela argentina hipotética, donde cada alumno tendrá una computadora, ésta podrá ser muy útil en la aplicación de experimentos, correcciones, etc. Para el primero se utiliza la técnica de simulación. Esta, como su nombre lo indica, consiste en simular un hecho real.

Por ejemplo, si una clase consiste en explicar tiro horizontal, trayectoria de un proyectil en el vacío, utilizar las posibilidades gráficas de la C-16 para simular el recorrido de una bala disparada por un cañón analizando los aspectos físicos del experimento, sería un hecho muy positivo dado que el educando experimenta directamente este acontecimiento fijando aun más los conceptos. Que él pueda descubrir cuál es el ángulo de inclinación para el cual se logra el alcance máximo, también representa un hecho importante. Más aún si lo comprueba teóricamente.

Otro caso que podemos citar es la autoevaluación; un examen tomado por la computadora. Cada pregunta no contestada implicaría un repaso inmediato del alumno utilizando, una vez más, la computadora.

En el área de lenguas se podría tomar, por ejemplo, una redacción y al final contabilizar la cantidad de errores de ortografía y/o puntuación. Luego indicarle al alumno, en el mismo momento, cuáles fueron sus errores dándole, también, cada una de las reglas de ortografía y puntuación. Imaginen por un momento una maestra dictando a sus alumnos. Ellos en vez de usar el clásico cuaderno utilizan el teclado de la C-64 o de la C-16 viendo lo que tipean sobre la pantalla. Luego de finalizar el dictado, cada alumno oprimirá una determinada tecla, la cual iniciará la evaluación. Cada error de ortografía será impreso en forma destellante, explicándole, además, por qué es una falta.

Prácticamente es posible utilizar estos dos equipos para explicar cualquier materia. Por ejemplo, en geografía, se puede utilizar los efectos gráficos de la C-16 para representar la zona en cuestión. De acuerdo a la tecla que se oprima, la computadora imprimirá las características climáticas y/o geográficas, realizando una ampliación de la zona de estudio.

En química, qué mejor que simular el efecto de una combinación entre dos sustancias sin correr ningún tipo de riesgo.

Para el sector contable también hay aplicaciones. Por ejemplo los mismos alumnos podrían desarrollar, o mejor dicho simular, la estructura y organización de una empresa compuesta por un determinado staf. Mantener actualizado el archivo de personal para luego liquidar sus sueldos. Esto implicaría calcular todos los aportes (jubilación, obra social, etc.) Además, podrían determinar, a través del mismo programa, los costos de posibles aumentos de producción.

En dibujo sería una herramienta muy interesante para diseñar perfiles a través del lápiz óptico y luego utilizar la computadora para ver si la estructura es la correcta.

Para historia también hay una aplicación. Podemos utilizar una base de datos junto con una C-64 y el drive 1541, para administrar los hechos históricos que se estudien.

Cuando el alumno tenga que estudiar un determinado tema, sólo deberá ingresar el título. De esta manera éste se iniciará realizando, además, pequeños test a lo largo del estudio.

Por ejemplo, la computadora imprimirá un determinado texto para luego preguntarle al estudiante algunos puntos sobre éste.

Para las materias más avanzadas, como puede ser electrónica o electricidad, es posible diseñar un software educativo que le permita al alumno diseñar esquemas eléctricos. De esta manera no se corre el riesgo de un corto circuito. Este sólo se imprimirá en pantalla.

Actualmente hay un utilitario para la C-64 (también para la C-128) que permite

diseñar un circuito lógico (escribir cada compuerta en pantalla) y, luego, ¡enviar cada combinación por la port del usuario!!!.
A través de todas estas prácticas el alumno podrá mejorar y desarrollar su lógica que, como se sabe, actualmente no se contempla en el sistema educativo nacional.
Claro que, para que se puedan lograr todas estas tareas, los docentes deben perfeccionarse y dominar esta nueva ciencia.
Como dijimos antes, es necesario que la educación se adecue al contexto que la rodea. Si hoy, la computadora ha invadido nuestros hogares, no se puede estar inmóvil y ser, nada más que simples usuarios de ella. Por el contrario, ella debe ser una herramienta más y no ser nosotros una herramienta de ella.
Queremos mencionar, para no dejarlos de lado, a los sistemas expertos capaces de "dominar" una determinada profesión. Mucho es lo que se está trabajando en estos sistemas, obteniéndose resultados increíbles.
Pueden perfectamente trabajar como el mejor médico clínico, abogado, etc.

¿Cómo sería el desempeño de un sistema experto en el área educativa? ¿Será ésta la educación del futuro? ¿Una computadora, en reemplazo del docente, enseñando a seres humanos?. Inmediatamente nuestra reacción refleja el "no es posible". Sin embargo 10 años atrás, las compañías automotrices de los Estados Unidos utilizaban operarios para armar cada una de las partes del producto. Hace una década ellos también dijeron "no es posible" cuando alguien les comentó que se estaba experimentando con unos raros aparatos que podían hacer todas las tareas que se les ordenaba. Hoy, en la misma empresa, se utilizan robots en lugar de aquellos operarios!!!

SOFTWARE EDUCATIVO PARA LOS EQUIPOS DREAN COMMODORE:

Existen varios paquetes de software desarrollados para la C-16 y C-64 dentro de este rubro. Para el área matemática hay un programa que nos permite practicar operaciones básicas de sumas, restas, división y multiplicación. Es ideal para que los niños realicen ejercicios. Otra de las ventajas es que se puede seleccionar el nivel de las operaciones. De esta manera puede ser utilizado por cualquier alumno de la escuela primaria.
Otro software desarrollado es el de temas monográficos. Aquí se explica paso a paso el funcionamiento de un determinado dispositivo. Por supuesto que para explicar un tema se utilizan todos los efectos audiovisuales de los equipos. Si se comenta, por ejemplo, como funciona el motor a explosión, se representa el movimiento de los pistones, el paso de combustible, etc.
No debemos dejar de mencionar el lenguaje (ya famoso) LOGO. Desarrollado por el Instituto de Tecnología de Massachusetts es actualmente el más utilizado para la enseñanza, especialmente de los niños. Es posible encontrar en nuestro país versiones de LOGO para la C-64 (tanto en castellano como en inglés).
Los seguidores de este lenguaje lo consideran como uno de los pocos que permite desarrollar más profundamente el intelecto de los educandos.

TRUCOS

FIGURAS ANIMADAS

Este pequeño programa crea un sprite que se mueve por la parte superior de la pantalla. Cuando el programa termina el movimiento de aquel, se imprime en pantalla para que lo puedas analizar. Cada línea viene acompañada por una sentencia REM seguida del comentario respectivo.

```
10 REM SPRITE EXPLICADO
20 POKE53281,6
30 FORS=832TO894: READT:
   POKES, T: NEXT
40 V=53248: POKE2040,13: REM
   AREA DEL SPRITE
50 POKEV+21,1: REM
   IMPRIMIMOS EL SPRITE
60 POKEV+39,1: REM COLOR
   BLANCO
70 POKEV+1,60: REM
   POSICION EJE Y
80 FORJ=1 TO255: POKEV,J:
   NEXTJ: REM POSICION EJE X
90 LIST
91 DATA0,0,0,64,0,0,64
92 DATA0,0,96,0,0,96,0,3
93 DATA252,0,0,240,0,0,96,1
94 DATA255,255,204,63,255,112,1
   24,240
95 DATA192,248,251,0,112,252,0,2
   255
96 DATA248,0,1,252,0,3,102,0
97 DATA28,99,0,120,97,192,248,96
98 DATA0,176,112,0,32,0,0,0
```

BLOCKS FREE

El siguiente programa les permitirá leer la cantidad de bloques libres que queden en el drive 1541. Recuerden que cada block de memoria equivale a 256 bytes.
Este programa lo pueden utilizar como una subrutina que determine la cantidad de bloques libres antes de proceder a grabar un archivo que pueda exeder la capacidad de almacenamiento actual.

```
10 LO=250: HI=2
20 Z$=CHR$(0)
30 OPEN15,8,15
40 PRINT # 15, "M-R"CHR$ (LO)
   CHR$(HI) CHR$(4)
50 GET # 15, LO$, L1$, HO$, H1$
60 FO=ASC (LO$ + Z$) + 256 *
   ASC (HO$ + Z$)
70 PRINT"BLOCKS FREE"; FO
80 CLOSE15
```

DEFAULT

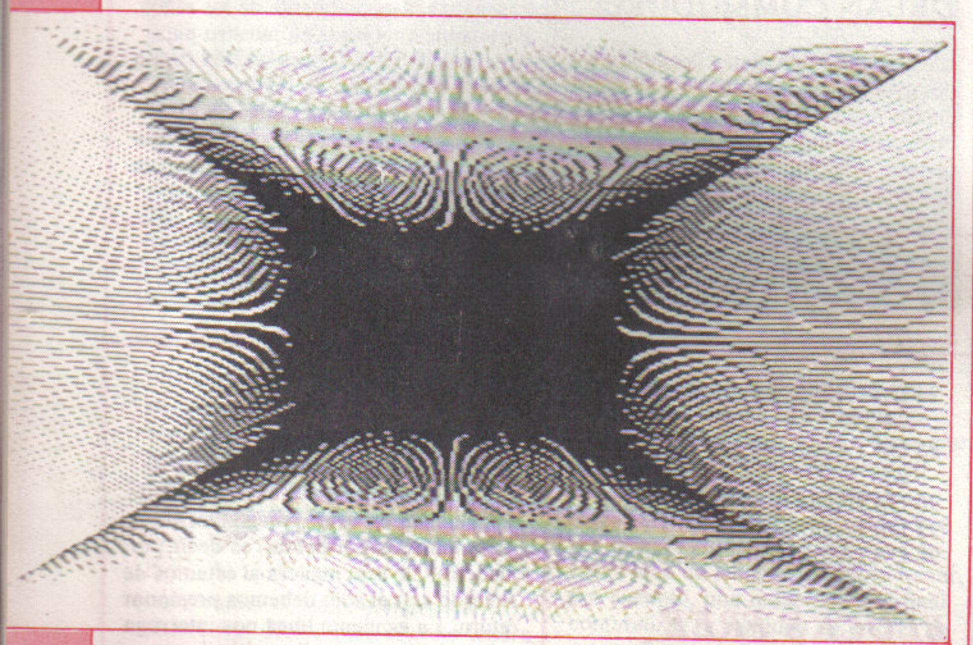
Tanto en los juegos como en los utilitarios, se nos pregunta acerca de determinadas opciones. Generalmente éstas tienen un valor preestablecido (denominados por defaul, es decir por defecto). De esta manera si estamos de acuerdo con él sólo debemos presionar return. La siguiente línea nos interroga sobre el número de dispositivo contestando, por default, el número ocho:
INPUT"NUMERO DE DISPOSITIVO[2 espacios]8[3 cursor iz.]";D
Las abreviaturas [2 espacios] y [3 cursor iz.] se deben entender como oprimir la tecla de espacio dos veces y oprimir la tecla que mueve el cursor hacia la izquierda tres veces respectivamente. De esta manera aparecerá el número ocho bajo el cursor.

FAST LOAD

Para aquellos que tengan el FAST LOAD les mostramos otra forma de activarlo. De acuerdo al fabricante, la única manera de lograrlo es reseteando la máquina. Esto también se logra realizando SYS57194.

CARACTERES Y GRAFICOS

A diferencia de la C-64, con la que la realización de buenos gráficos resulta una tarea tediosa, la 16 dispone de comandos de fácil manejo que permiten cómodamente crear las más diversas figuras.



Una de las principales características de la Drean Commodore 16 es la posibilidad de realizar excelentes gráficos en alta resolución. A diferencia de la C-64, en donde la realización de estos gráficos demanda ingresar valores predeterminados en los registros respectivos de video, resultando una tarea muy tediosa, la C-16 dispone de comandos de fácil manejo que permiten cómodamente crear las más diversas figuras.

Podemos igualmente desarrolllar gráficos sin necesidad de utilizar estos comandos.

Si observan las teclas de la C-16 verán que cada una de ellas tiene tres caracteres: la letra en sí y dos caracteres gráficos. Utilizando el comando PRINT en todas sus variantes juntamente con estos dos caracteres podremos diseñar gráficos que sirvan como escenarios para nuestros primeros juegos.

Para acceder a cada uno de esos dos caracteres debemos presionar la tecla SHIFT o COMMODORE según sea el caso.

Si presionamos SHIFT y la tecla deseada, lograremos imprimir el caracter gráfico de la derecha.

Por el contrario la tecla COMMODORE y la tecla respectiva imprimen el caracter de la izquierda.

¿Cómo interviene el comando PRINT en la creación de los gráficos? Para ello debemos recordar los formatos del comando.

Para imprimir leyendas o comentarios se utiliza el PRINT seguido del mensaje encerrado entre comillas.

Por ejemplo PRINT"**********" imprimirá sobre la pantalla diez asteriscos.

El segundo formato es el PRINT TAB(x) el cual imprimirá una serie de caracteres a partir de la columna especificada por x (x debe ser un número natural comprendido entre 1 y 40).

Un ejemplo de ello sería PRINT TAB(19)"**********" el cual realizará lo mismo que en el caso anterior con la salvedad que comenzará a imprimirlos a partir de la columna19.

El tercer formato es sumándole un ";" al final del comando. De esta manera no se realizará retorno de carro. Es decir que cuando se encuentre otro PRINT, éste se imprimirá a continuación del anterior.

Por ejemplo
PRINT"**********";
PRINT"**********" imprimirá veinte asteriscos en la misma línea. El último formato es igual que el anterior pero utiliza una coma (,). El efecto que se produce es la separación entre un PRINT y el que sigue.

Para empezar podemos utilizar todos estos formatos juntamente con los caracteres gráficos. Es una muy buena manera de introducirnos en la creación de figuras.

Si queremos "tecnificar" nuestros gráficos podemos utilizar las posibilidades de imprimir ciertos caracteres en video inverso.

Esto significa que si nosotros tenemos el cursor negro y el fondo amarillo, todo lo que se imprima en este modo aparecerá en letras amarillas y en fondo negro.

Esto se logra utilizando RVS ON y RVS OFF. Cada uno de éstos se logra presionando la tecla de control (CTRL) y, simultáneamente, la tecla que contiene al número 9 o la que contiene al número 0 (observen que debajo de éstas están las abreviaturas antes descriptas).

Para ejemplificar lo último dicho tipeen en modo directo: PRINT"[RVS ON] PRUEBA DE VIDEO INVERSO [RVS OFF]" y opriman return.

Observen luego lo que sucede (nota: [RVS ON significa que se debe oprimir la tecla de CTRL y la tecla que contiene al número 9 simultáneamente; mismo caso para RVS OFF). Más fácil aún, opriman [RVS ON] directamente y comiencen a tipear normalmente.

Otro accesorio interesante es el

FLASH. Este se acciona en forma similar que el anterior pero oprimiendo "<" o ">" según sea el caso. Con FLASH ON comienzan a parpadear todos los caracteres que se encuentran desde el comando hasta el FLASH OFF.

Podemos seguir agregándole todos los detalles que querramos.

Si deseamos que cada gráfico tenga un color diferente debemos seleccionarlo oprimiendo la tecla de CTRL o C= (tecla de commodore) juntamente con el color seleccionado. Además de hacer esto en modo directo, podemos utilizarlo con una sentencia PRINT, como por ejemplo PRINT"|CTRL 3| ESTO ES ROJO" se imprimirá, justamente, en rojo (nota: la abreviatura |CTRL 3| significa que se debe oprimir la tecla CTRL y el 3).

Antes de continuar avanzando en el diseño de gráficos, es muy importante practicar todas las posibilidades que aqui les mencionamos. Les sugerimos que agoten todas las formas de crear gráficos utilizando las herramientas descriptas hasta aqui.

Gráficos avanzados:

Comentaremos ahora los comandos que tiene la C-16 para el diseño de gráficos. Les daremos como ejemplo algunos listados, los cuales realizan unos gráficos especiales.

Para iniciar todo gráfico debemos "conmutar" la Drean Commodore 16 a modo gráfico. Esto se efectúa usando el comando GRAPHIC.

De esta manera le indicamos a la computadora si trabajaremos con texto, en alta resolución, texto y gráficos de alta resolución, gráficos multicolores o con gráficos multicolores con texto.

Si consultan su manual verán que el formato de este comando es GRAPHIC modo,borrado (opcional).

El modo es un número entre 0 y 4 correspondiendo a los recién descriptos. El parámetro 'borrado' está comprendido entre 0 y 1 permitiendo, de acuerdo al valor, borrar o no la pantalla antes de comenzar a graficar. Nosotros como trabajaremos en alta resolución utilizaremos GRAPHIC 1,0. Los siguientes listados realizan diversas figuras. Cada uno de ellos utiliza el modo de alta resolucion:

```
10 REM GRAFICO NRO 1
30 A$="C"
40 DO WHILE A$ <> "F"
50 GRAPHIC1,1
60 BG=INT(RND(1)*16+1)
70 FG=INT(RND(1)*16+1):IF
   BG=FG THEN 60
80 S=INT(RND(1)*4+3)
90 COLOR0,BG:COLOR1,FG
100 FOR I=0 TO 199 STEP S
110 Y=199-I
120 DRAW1,0,I TO 319,Y
130 NEXT I
140 FOR I=319 TO 0 STEP -S
150 DRAW 1,I,0 TO 319-I,199
160 NEXT I
170 GET A$: IF A$="" THEN 170
180 LOOP
190 GRAPHIC 0
200 COLOR 0,7: COLOR 1,1
210 END
```

Este programa realiza una figura similar a una interferencia radioeléctrica. Para finalizar la ejecución, deben oprimir la tecla F. Oprimiendo cualquier otra se cambiará el color de la figura. Este programa, además, utiliza sentencias orientadas a la programación estructurada como es el caso de WHILE.

Les sugerimos que carguen y ejecuten este programa para luego modificar los valores en las sentencias DRAW. De esta manera podrán experimentar con los gráficos en alta resolución.

```
20 REM GRAFICO NRO 2
25 A$="C"
30 DO WHILE A$ <>"F"
40 GRAPHIC 3,1
41 BG=INT(RND(1)*16+1)
42 FG=INT(RND(1)*16+1): IF
   BG=FG THEN 41
43 M1=INT(RND(1)*16+1): IF
   M1=FG OR M1=BG THEN 43
44 M2=INT(RND(1)*16+1): IF
   M2=FG OR M2=BG OR
   M2=M1 THEN 44
45 COLOR 0,BG: COLOR 1,FG:
   COLOR 2,M1: COLOR 3,M2
50 FOR N=1 TO 40
60 FOR C=1 TO 3
70 CIRCLE C,50,87,N
80 CIRCLE C,89,87,N
90 NEXT C
100 NEXT N
110 GET A$: IF A$="" THEN 110
120 LOOP
130 GRAPHIC 0
140 END
```

Ese programa realiza dos circunferencias concéntricas. Igual que el programa anterior, el programa finalizará cuando se oprima la tecla F. Si se oprime cualquier otra se cambia el color del gráfico. Variando los valores de las líneas 70 y 80 se obtendrán otro tipo de figuras, también en alta resolución.

DATABASE: BASE DE DATOS PARA LA C-64

El siguiente programa utilitario les permitirá crear su propia base de datos, manipulando cómodamente toda la información.

```
CREAR UN ARCHIVO
AGREGAR REGISTROS AL ARCHIVO
MODIFICAR LOS REGISTROS DEL ARCHIVO
BORRAR REGISTROS DEL ARCHIVO
LEER EL ARCHIVO DESDE EL DISCO
VER EL ARCHIVO SOBRE PANTALLA
ORDENAR EL ARCHIVO
ESCRIBIR EL ARCHIVO EN DISCO
$ DIRECTORIO DEL DISCO
SALIR DEL PROGRAMA
OPRIMA LA TECLA APROPIADA
```

Son varias las definiciones sobre bases de datos. Algunos autores las representan como una serie de programas que permiten manipular cómodamente los datos. Otros las definen como una colección de datos mutuamente relacionados, al hardware de la computadora que se emplea para almacenarlos y a los programas que se utilizan para manipularlos.

Seguramente, esta es la definición más completa de una base de datos.

¿Para qué sirve o cómo se puede utilizar una base de datos? La respuesta será mejor darla a través de un ejemplo. Supongamos que queremos desarrollar una agenda personal, cuyos datos están formados por nombre, apellido, dirección, código postal y teléfono.

Para trabajar mejor, ordenaremos la agenda alfabéticamente (por apellido). De esta manera por cada nuevo dato que insertemos, estaremos seguros que su ubicación dentro de la agenda es la correcta.

Además si al buscar un determinado dato éste no se encuentra, también estaremos seguros de que nunca ha sido escrito por nosotros, salvo que se haya producido un error en el momento del asentamiento dentro de la agenda (es decir ponerlo en el lugar que no le corresponde).

Una base de datos realiza todas estas operaciones. Por supuesto permite realizar mucho más. Por ejemplo, podemos ordenar la agenda por nombre o por número de teléfono, o por código postal. No hace falta mencionar la velocidad de ejecución y la confiabilidad del proceso.

Antes de comenzar a describir qué puede hacer DATABASE, repasemos algunos conceptos.

Cada base de datos, al igual que los archivos, está constituida por registros. A su vez cada uno de ellos está formado por campos.

En el ejemplo de la agenda, cada dato es un registro. Los campos son el nombre, apellido, dirección, código postal y teléfono (ver fig. 1).

En esta base de datos, denominada AGENDA, tenemos tres registros cada uno con sus correspondientes campos. Como ven el listado no está escrito en un determinado orden. Una de las habilidades de la base de datos es poder ordenarlos acorde al campo que nosotros seleccionamos. Por ejemplo si hubiésemos ordenado nuestra base por

nombre, el resultado hubiese sido 1, 2 y 3. En cambio si se ordena respecto a la dirección: 2, 3, 1.
DATABASE es una base de datos desarrollada para la Drean Commodore 64 juntamente con el disk drive 1541. Ocupa un total de memoria de aproximadamente 7 Kb, estando totalmente escrito en lenguaje Basic.
Las operaciones que permite realizar son:
AGREGAR un registro
MODIFICAR los campos de los registros
BORRAR registros del archivo
VER el archivo en pantalla
ORDENAR el archivo
BUSCAR datos dentro del archivo
Para no crear confusiones, aquí impondremos la palabra 'archivo' como sinónimo de base de datos.

Descripción

Una vez que hayamos cargado el programa en memoria, y luego de tipear RUN, aparecerá en pantalla el siguiente menú:

DATABASE
CREAR UN ARCHIVO
AGREGAR REGISTROS AL ARCHIVO
MODIFICAR LOS REGISTROS DEL ARCHIVO
BORRAR REGISTROS DEL ARCHIVO
LEER EL ARCHIVO DESDE EL DISCO
VER EL ARCHIVO SOBRE PANTALLA
ORDENAR EL ARCHIVO
ESCRIBIR EL ARCHIVO EN DISCO
$ DIRECTORIO DEL DISCO
SALIR DEL PROGRAMA

Como verán, cada una de las primeras letras de cada función están impresas en video inverso. De esta manera, es decir oprimiendo la deseada, se seleccionará la función.
Primeramente, debemos definir los registros que constituirán nuestro archivo.
Esto lo realiza la primera opción: crear un archivo. Luego de oprimir la tecla C, y si ya no hemos definido anteriormente otro archivo, se nos preguntará la cantidad de campos que tendremos en cada registro.
Siguiendo con el ejemplo de la agenda, debemos responder 5 (nombre, apellido, dirección, CP y TE).
A continuación se nos interrogará por el título y la longitud de cada campo. Esta se refiere a la cantidad máxima de caracteres que tendrá.
En nuestro caso debemos contestar:

CAMPO NRO 1
TITULO ? NOMBRE
LONGITUD ? 10

CAMPO NRO 2
TITULO ? APELLIDO
LONGITUD ? 10

CAMPO NRO 3
TITULO ? DIRECCION
LONGITUD ? 15

CAMPO NRO 4
TITULO ? CP
LONGITUD ? 4

CAMPO NRO 5
TITULO ? TE
LONGITUD ? 8

De esta manera indicamos cómo están formados los registros del archivo.
Luego de ingresar los datos del último campo, DATABASE nos dirá, de

acuerdo a esos valores, cuántos registros podemos tener.
De no estar de acuerdo con el total máximo, se nos permite redefinir la longitud y/o título de cada campo (nota: a menor cantidad de caracteres por registro, mayor será la cantidad de registros que podremos tener).
En caso de aceptar la máxima cantidad, el control retornará al menú principal.
A partir de aquí podemos comenzar a ingresar los datos. Esto se realiza a través de la opción A (agregar registros al archivo). Aquí se imprimirá cada uno de los campos antes definidos. Es decir:

NOMBRE: CARLOS (reg. 1)
APELLIDO: ROMANO
DIRECCION: TANDIL 1221
CP: 1670
TE: 778-2212

NOMBRE: GABRIEL (reg. 2)
APELLIDO: CASTRO
DIRECCION: JUJUY 210
CP: 1090
TE: 855-5590

NOMBRE: RUBEN (reg. 3)
APELLIDO: TORRISI
DIRECCION: NEWTON 921
CP: 1281
TE: 22-2112

(fig. 1)

La entrada se finaliza, como se indica, presionando la tecla RETURN. Así se regresa al menú principal.
Las opciones de borrar y modificar los registros tienen sus propias instrucciones. Básicamente piden el número de registro a borrar (modificar) o, en el otro caso, la letra T la cual significa que se borrarán (modificarán) todos los registros del archivo.
Leer el archivo desde el disco y escribir el archivo en disco se refieren a las operaciones LOAD y SAVE. De esta manera podemos almacenar nuestra agenda para luego utilizarla.
La opción O (ordenar el archivo) clasifica el archivo acorde al campo seleccionado. Por este motivo se nos interrogará con respecto a qué campo se ordenará la base.
Siguiendo en nuestro ejemplo aparecerá:
1 NOMBRE
2 APELLIDO
3 DIRECCION
4 CP
5 TE

Por qué campo ordena el archivo?

Aquí debemos ingresar el número de campo deseado (si ordenamos la base por dirección, el número a ingresar es el 3). Una aclaración importante es que la clasificación se hará ascendentemente (de menor a mayor). Una vez que la clasificación finalice, el control se transferirá al menú principal.
Por último describiremos la opción V correspondiente al ítem "ver el archivo sobre pantalla".
Luego de oprimir la tecla V, el programa nos imprimirá (siempre en pantalla) el primer registro del archivo.
Seguidamente nos interrogará por la operación que deseamos hacer.
El ítem V se subdivide en otras operaciones. Estas son PROXIMO, ANTERIOR, SALTAR, BUSCAR, VOLVER AL MENU.
Al igual que en el menú principal, la primera letra de cada operación está impresa en video inverso. Estas determinan la operación a efectuarse. Si se oprime la "P", se imprimirá el próximo registro. Si se oprime la "A" se imprimirá el registro anterior al actual. Con "S" podemos ver cualquier registro que esté dentro del rango del archivo. Por ejemplo, para ver el registro número 25, evitando acceder a él pasando antes por los 24 anteriores, utilizamos esta opción. Aquí, simplemente, debemos ingresar el número de registro respondiendo, así, a la pregunta NUMERO DE REGISTRO AL CUAL SALTA.
La última opción del ítem "V" es buscar. A través de ella podemos saber si un determinado dato se encuentra en la base. Al igual que en clasificación, la búsqueda se puede realizar por campo. Sólo se pide un requisito: que el archivo esté, antes de iniciar la búsqueda, ordenado acorde a ese campo. Esto se debe a que el algoritmo de búsqueda funciona correctamente cuando el archivo está ordenado. De otra manera puede ocurrir que un dato que sí se encuentra en la base no sea hallado

```
10 REM DATABASE
12 POKE53280,13:POKE53281,11:POKE650,128:PRINT"■":GOSUB16:IFX=0THENGOTO66
14 GOTO68
16 D$=CHR$(0):MR$=D$:DR$=D$:S=0:B1$=CHR$(10):PW=0:CW=0:B$=CHR$(32)
18 NC=0:PG=0:NL=0:F1=0:F2=0:F3=0:L$=D$:RL=0:SB$=D$:CR$=CHR$(13):HN$=D$:ID$=D$
20 A$=D$:C$=D$:T%=0:I$=D$:CK=0:I=0:J=0:K=0:L=0:M=0:N=0:RW=5:SF=0:Z=0:E$="EOF"
22 MEM=31690:EN=0:EM$=D$:ET=0:ES=0:A1$=D$:A2$=D$:A3$=D$:RETURN
24 DIMF$(F+1),T%(F+1),L%(F+1):RETURN
26 DIM REC$(R+1,F+1),ML$(9,4),PC(10),TT$(5),HC$(9),K%(R+1):RETURN
28 REM---GET---
30 GETA$:IFA$=""THEN30
32 RETURN
34 REM---CREACION---
36 IFCK<>0THENGOSUB394
38 PRINT"■ ■      INICIALIZANDO DATABASE          ":PRINT:PRINT
40 CLR:GOSUB16:INPUT"■CUANTOS CAMPOS EN CADA REGISTRO? 0 ■■■■■";F:IFF=0THEN68
42 GOSUB24:FORI=1TOF
44 PRINT"■■■■■■CAMPO NRO";I:PRINT"■TITULO ?                              "
46 PRINT"LONGITUD ?                     ■"
48 PRINT"■■■■■";TAB(7);:INPUTF$(I):PRINT TAB(9);:INPUTL%(I):NEXTI
50 REM---DETER NRO.DE REGISTROS---
52 FORJ=0TOF:RL=RL+L%(J):NEXTJ:RL=RL+3*(F+1)+5:R=INT((MEM-12*(F+1)-2100)/RL)
54 PRINT"■SU SELECCION LE PERMITIRA TENER APROX"
56 PRINTR;"REGISTROS.  ■A■CEPTA O  ■R■EINGRESA?"
58 GOSUB30:IFA$="R"THENPRINT"■":GOTO38
```

```
60 IFA$="A"THENGOSUB26:CK=1:GOTO68
62 GOTO58
64 REM---MENU---
66 PRINT"] ■               DATABASE              ■ ":GOTO70
68 PRINT"]":PRINT" ■               MENU PRINCIPAL              "
70 PRINT"■  ■C■REAR UN ARCHIVO"
72 PRINT"■  ■A■GREGAR REGISTROS AL ARCHIVO"
74 PRINT"■  ■M■ODIFICAR LOS REGISTROS DEL ARCHIVO"
76 PRINT"■  ■B■ORRAR REGISTROS DEL ARCHIVO"
78 PRINT"■  ■L■EER EL ARCHIVO DESDE EL DISCO"
82 PRINT"■  ■V■ER EL ARCHIVO SOBRE PANTALLA"
84 PRINT"■  ■O■RDENAR EL ARCHIVO"
86 PRINT"■  ■E■SCRIBIR EL ARCHIVO EN DISCO"
88 PRINT"■  ■$■ DIRECTORIO DEL DISCO"
89 PRINT"■  ■S■ALIR DEL PROGRAMA"
90 PRINT" ■         OPRIMA LA TECLA APROPIADA       "
92 PRINT"         HAY";X;"REGISTROS EN MEMORIA"
94 IFR>0THENPRINT"        ESPACIO PARA";R-X;"REGISTROS MAS";
96 GOSUB30:IFA$="A"THENGOSUB350:GOTO124
98 IFA$="M"THENGOSUB350:GOTO243
100 IFA$="B"THENGOSUB350:GOTO272
102 IFA$="C"THEN36
104 IFA$="L"THEN170
108 IFA$="V"THENGOSUB350:GOTO192
110 IFA$="E"THENGOSUB350:GOTO144
112 IFA$="O"THENGOSUB350:GOTO304
114 IFA$="S"THEN342
118 IFA$="$"THEN 422
120 GOTO96
122 REM---AGREGA REGISTROS---
124 FORI=X+1TOR:PRINT"]OPRIMA LA TECLA ■RETURN■ LUEGO DE CADA EN--TRADA."
126 PRINT"OPRIMA SOLAMENTE ■RETURN■ CUANDO DESEE    FINALIZAR LA ENTRADA.■■"
128 PRINT"■REGISTRO NUMERO ";I;"■"
130 FORN=1TOF
132 PRINTF$(N);"       >■■■";:INPUTREC$(I,N):IFREC$(I,N)=""THENREC$(I,N)=">"
134 IFLEN(REC$(I,N))>L%(N)THENGOSUB140:GOTO132
136 IFREC$(I,1)=">"THENX=I-1:CK=1:GOTO68
138 NEXTN:K%(I)=I:NEXTI:X=R:CK=1:GOTO68
140 PRINT"NO PUEDE EXEDER LOS ■";L%(N);" CARACTERES.":RETURN
142 REM---SAVE---
144 PRINT"]■INGRESE EL NOMBRE DEL ARCHIVO QUE SERA  GRABADO ";
146 PRINT"(12 CARACTERES MAX). SI EXISTE  UN ARCHIVO ";
148 PRINT"CON EL MISMO NOMBRE, ESTE     SERA BORRADO.■■"
150 PRINT"  ";NF$:INPUT"]";NF$:IFNF$=""THEN68
152 OPEN 15,8,15:PRINT#15,"S0:DF] "+LEFT$(NF$,8)+"!OLD":GOSUB414
154 PRINT#15,"R0:DF] "+LEFT$(NF$,8)+"!OLD=DF] "+NF$:GOSUB414
156 OPEN 5,8,5,"0:DF] "+NF$+",S,W":GOSUB414
158 PRINT#5,R;CR$;F;CR$;X:GOSUB414:FORN=1TOF:PRINT#5,F$(N);CR$;L%(N):NEXTN
160 FORI=1TOX:PRINT"■GRABANDO REGISTRO NRO";I;"]]"
162 FORN=1TOF:PRINT#5,REC$(I,N):NEXTN:GOSUB414:NEXTI:PRINT
164 FORI=1TOX:PRINT"■GRABANDO PUNTERO NRO";I;"]]":PRINT#5,K%(I):NEXTI:GOSUB414
166 PRINT#5,E$:CLOSE5:CLOSE15:CK=0:GOTO68
168 REM---LOAD---
170 IFCH<>0THENGOSUB394
172 CLR:GOSUB16:PRINT"]INGRESE EL NOMBRE DEL ARCHIVO QUE SERA  LEIDO■■":INPUTNF:
174 OPEN15,8,15:OPEN5,8,5,"0:DF] "+NF$+",S,R":GOSUB414
176 INPUT#5,R,F,X:GOSUB414:GOSUB24:GOSUB26:FORN=1TOF:INPUT#5,F$(N),L%(N):NEXTN
178 FORI=1TOX:PRINT"■LEYENDO REGISTRO NRO";I;"]]"
180 FORN=1TOF:INPUT#5,REC$(I,N):NEXTN:GOSUB414:NEXTI:PRINT
182 FORI=1TOX:PRINT"■LEYENDO PUNTEROS";I;"]]":INPUT#5,K%(I):NEXTI
184 GOSUB414:S=ST:IFS<>0THEN188
186 INPUT#5,E$
188 CLOSE5:CLOSE15:GOTO68
190 REM---MUESTRA REG---
192 I=1
```

```
194 IFI=0THEN68
196 IFI>XTHEN68
198 PRINT"■■■■REGISTRO NUMERO■";I;" ■EN ARCHIVO■ ";NF$;"■■"
200 FORN=1TOF:PRINT F$(N);": ■";REC$(K%(I),N):NEXTN
202 PRINT"■ ■P■ROXIMO, ■A■NTERIOR, ■S■ALTAR, ■B■USCAR.      ■V■OLVER AL MENU."
204 GOSUB30:IFA$="P"THENI=I+1:GOTO194
206 IFA$="A"THENI=I-1:GOTO194
208 IFA$="S"THEN216
210 IFA$="B"THEN500
212 IFA$="V"THEN68
214 GOTO204
216 INPUT"■NUMERO DEL REGISTRO AL CUAL SALTA ";I:GOTO194
242 REM---MODIFICA---
243 PRINT"■■MODIFICA ■T■ODOS O UN SOLO REGISTRO ";
244 PRINT"(EN ESTE CASO INGRESE EL NUMERO DEL REGISTRO)":INPUTMR$:IFMR$=D$THEN68
245 PRINT"■■"
246 IFMR$="T"THENMR$=D$:GOTO254
248 I=VAL(MR$):MR$=D$
250 IFI>XTHENGOSUB348:GOTO244
252 GOSUB256:GOTO68
254 FORI=1TOX:GOSUB256:NEXTI:GOTO68
256 PRINT"■■PARA MODIFICAR EL REGISTRO NUMERO";I;", REALIZE LOS CAMBIOS ";
258 PRINT"CUANDO CADA CAMPO SEA IMPRESO. LUEGO OPRIMA ■RETURN■■"
260 FORN=1TOF:PRINTF$(N)":":PRINT"    ■";REC$(K%(I),N)
261 IFLEN(REC$(K%(I),N))>36THENPRINT"■";
262 PRINT"■ ";:INPUTREC$(K%(I),N)
264 IFLEN(REC$(K%(I),N))>L%(N)THENGOSUB140:GOTO260
266 IFREC$(K%(I),N)=""THENREC$(K%(I),N)=">"
268 NEXTN:CK=1:RETURN
270 REM---BORRA---
272 PRINT"■■BORRA ■T■ODOS LOS REGISTROS O UNO SOLO ";
273 PRINT"(ENESTE CASO INGRESE EL NUMERO DE REGISTRO) ■■"
274 INPUTDR$:IFDR$=D$THEN68
276 IFDR$="T"THENDR$=D$:GOTO282
278 I=VAL(DR$):DR$=D$:IFI>XTHENGOSUB348:GOTO274
280 GOSUB284:GOTO68
282 FORI=1TOX:GOSUB284:NEXTI:GOTO68
284 PRINT"■■PARA BORRAR EL REGISTRO NRO";I;", OPRIMA"
286 PRINT"■SHIFT■ ■D■, OPRIMA LA TECLA ESPACIADORA    PARA AVANZAR■"
288 FORN=1TOF:PRINTF$(N);"    ■";REC$(K%(I),N):NEXTN
290 GOSUB30:IFASC(A$)=196THEN294
292 CK=1:RETURN
294 PRINT"■■BORRANDO REGISTRO";I:PRINT"■■YA NO SE ENCUENTRA EN EL ARCHIVO"
295 FORD=1TO1000:NEXTD
296 FORN=1TOF:REC$(K%(I),N)=REC$(X,N):REC$(X,N)="":NEXTN
298 FORJ=1TOX:IFK%(J)=XTHENK%(J)=K%(X):K%(X)=0:X=X-1:GOTO292
300 NEXTJ
302 REM---ORDENA---
304 PRINT"■■ ■  ORDENA LOS REGISTROS ASCENDENTEMENTE■■"
306 FORN=1TOF:PRINT" ■";N;"■ ";F$(N):NEXTN
308 INPUT"■POR QUE CAMPO ORDENA EL ARCHIVO? 0 ■■■■■";SF:IFSF=0THEN68
310 IFSF>FTHENPRINT"■■■":GOTO308
312 PRINT"■ ■UN MOMENTO ■■":M=X
314 M=INT(M/2):IFM=0THENCK=1:GOTO68
316 J=1:K=X-M
318 I=J
320 L=I+M
322 PRINT"ORDENANDO     ■■■■■■";I;"■"
324 IFREC$(K%(I),SF)<=REC$(K%(L),SF)THEN328
326 T%(N)=K%(I):K%(I)=K%(L):K%(L)=T%(N):I=I-M:IFI>0THEN320
328 J=J+1:IFJ>KTHEN314
330 GOTO318
332 REM---SALE---
334 PRINT"■■■ ■UD NO HA GRABADO SU ARCHIVO !"
336 PRINT"■■ DESEA REALMENTE SALIR DEL PROGRAMA ?    ■S■ O ■N■"
```

```
338 GOSUB30:IFA$="S"THEN344
340 GOTO68
342 IFCK<>0THEN334
344 PRINT"TTTFIN DE DATAFILE":END
346 REM---ERROR---
348 PRINT" NO EXISTE TAL REGISTRO":RETURN
350 IFR>0THENRETURN
352 PRINT"Ɔ▓ NO HAY ARCHIVO EN MEMORIA":FORI=1TO1000:NEXTI:GOTO68
354 IFX<1THENGOSUB352:GOTO68
356 RETURN
392 REM---CUIDADO---
394 PRINT"Ɔ▓ ▓ESTO DESTRUIRA EL ARCHIVO EN MEMORIA !"
396 PRINT"▓ ALMACENA, PRIMERO, EL ARCHIVO? ▓S■ O ▓N■":GOSUB30:IFA$="N"THENRETURN
398 GOTO68
412 REM---ERROR DE DISCO---
414 INPUT#15,EN,EM$,ET,ES:IF(EN<20)OR(EN=62)THENRETURN
416 PRINT"▓▓DISK ERROR■"EN"▮, "EM$","ET"▮,"ES
418 PRINT"▓OPRIMA ▓UNA TECLA■ PARA VOLVER AL MENU":GOSUB30:CLOSE5:CLOSE15:GOTO68
420 REM---DIRECTORIO---
422 OPEN 15,8,15:OPEN1,8,0,"$0":PRINT"Ɔ":GOSUB414
424 GET#1,A1$,A2$
426 GET#1,A1$,A2$
428 GET#1,A1$,A2$
430 IFA1$<>""THENA0=ASC(A1$)
432 IFA2$<>""THENA0=A0+ASC(A2$)*256
434 PRINTMID$(STR$(A0),2);TAB(3);
436 GET#1,A2$:IFST<>0THEN454
438 IFA2$<>CHR$(34)THEN436
440 GET#1,A2$:IFA2$<>CHR$(34)THENPRINT"▓"A2$"■";:GOTO440
442 GET#1,A2$:IFA2$<>CHR$(32)THEN442
444 PRINTTAB(20);:A3$=""
446 A3$=A3$+A2$:GET#1,A2$:IFA2$<>""THEN446
448 PRINTLEFT$(A3$,3)
450 GETA$:IFA$<>""THENGOSUB458
452 IFST=0THEN426
454 PRINT" BLOCKS FREE";:A0=0
456 CLOSE1:CLOSE15:PRINTTAB(20)"OPRIMA ▓UNA TECLA■":GOSUB30:GOTO68
458 GOSUB30:RETURN
500 REM---RUTINA DE BUSQUEDA---
510 PRINT"Ɔ               ▓ RUTINA DE BUSQUEDA ■"
515 PRINT
520 PRINT"PARA BUSCAR ALGUN REGISTRO ▓PRIMERO ORDE--NE■ EL ARCHIVO ";
530 PRINT"ACORDE AL CAMPO QUE SERA BUSCADO. ";
540 PRINT"DE OTRA MANERA, EL DATO . ▓NO PO--DRA SER HALLADO■.":PRINT
550 FORI=1TOF:PRINT" ▓";I;"■ ";F$(I):NEXTI
555 PRINT
560 INPUT"POR QUE CAMPO BUSCARA  0▮■■▮";SF:IFSF=0THEN68
561 IFSF>FTHEN560
570 INPUT"DATO A BUSCAR <▮■■▮";BU$
572 PRINT:PRINT" ▓ UN MOMENTO ■":PRINT
575 IFBU$="<"THEN68
580 UL=X+1:HE=1
590 DA=INT((UL+HE)/2)
595 PRINT"BUSCANDO           ▮▮■■▮";DA;"Ɔ"
600 IFREC$(K%(DA),SF)=BU$THEN640
610 IFHE=UL-1THEN690
620 IFBU$>REC$(K%(DA),SF)THENHE=DA:GOTO590
630 UL=DA:GOTO590
640 PRINT:PRINT:FORI=1TOF:PRINT" ▓";TAB(14);REC$(K%(DA),I):NEXTI
650 PRINT"                  ▓REGISTRO NRO■";DA
660 PRINT:PRINT:PRINT"  ▓C■ONTINUA O ▓V■UELVE AL MENU PRINCIPAL"
670 GOSUB30:IFA$="C"THEN510
680 IFA$="V" THEN68
682 GOTO670
690 PRINT"▓▓        ▓ EL DATO NO SE ENCUENTRA ■":GOTO660
```

MAPA DE MEMORIA

Tercera entrega de esta serie que, obviamente, no es para neófitos, ya que se requieren conocimientos del lenguaje máquina de la C-64.

Dirección $0043-$0044 (67-68):
Las sentencias GET, READ, e INPUT utilizan estas direcciones como un puntero a la dirección de entrada del dato fuente, como el caso de DATA, o de texto dentro del buffer de entrada en la dirección $200 (512).

Dirección $0045-$0046 (69-70):
Aquí se almacena el nombre de la variable basic que el intérprete basic está buscando. El formato que se utiliza para guardar el nombre es el mismo que se utiliza en la dirección $2D (45).

Dirección $0047-$0048 (71-72):
Estas direcciones apuntan al área de almacenamiento de variables, más precisamente a valor de la variable cuyo nombre está almacenado en las direcciones anteriores.
Cuando se ejecuta un FN, esta dirección no apunta a la variable independiente (la A de FN A, por ejemplo). De esta manera se evita el hecho de modificar el valor de esa variable cuando se ejecuta el FN.

Dirección $0049-$004A (73-74):
Aquí se almacena la variable que se utiliza en un lazo FOR-NEXT. Luego ésta se transfiere al stack.
Estas direcciones son utilizadas, también, como un área de trabajo por las sentencias INPUT, GET, READ, LIST, WAIT, CLOSE, LOAD, SAVE, RETURN y GOSUB.

Dirección $004B-$004C (75-76):
Estas direcciones son utilizadas durante la evaluación de una expresión matemáticas para determinar el desplazamiento necesario para tomar el operador desde una tabla.
Estas direcciones también se utilizan como un puntero auxiliar del texto basic.

Dirección $004D (77-78):
Aquí se almacena un valor el cual permite saber si el resultado de una comparación fue menor (1), igual (2) o mayor.

Dirección $004E-$0053 (78-83):
Estas direcciones se utilizan como punteros varios durante el proceso de definición y ejecución de funciones definidas por el usuario (DEF FN).

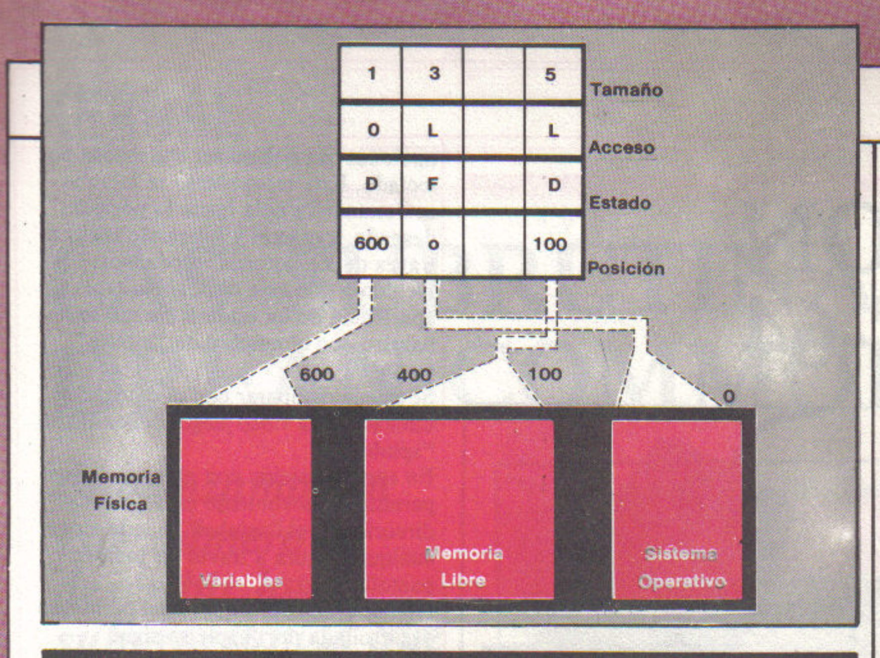
Dirección $0054- $0056 (84-86):
En estas tres direcciones se encuentran el salto hacia la función definida. En la primer dirección se encuentra el código de operación de la instrucción JMP ($4C). En la segunda y tercer dirección se halla la dirección de la función requerida.

Dirección $0057-$0060 (87-96):
Esta es un área de trabajo utilizada por varias rutinas.

Tabla 1

$0073	CHRGET	INC $7A	; incrementa el byte bajo del ;puntero
$0075		BNE CHRGOT	;salta a CHRGOT si no es cero
$0077		INC $7B	;si es cero incrementa el byte alto
$0079	CHRGOT	LDA $0207	;toma el byte desde el puntero ;(aqui desde el buffer de entrada)
$007C		CMP #$3A	;setea el flag de carry si el byte ;leido es > al ASCII del número 9
$007E		BCS FIN	;el caracter no es un número; ;finalizamos
$0080		CMP #$20	;si se leyó un espacio
$0082		BEQ CHRGET	;entonces tomamos el próximo ;caracter
$0084		SEC	;se prepara para la resta
$0085		SBC #$30	;los ASCII de 0-9 están entre ;$30-$39
$0087		SEC	;se prepara nuevamente para la ;resta
$0088		SBC #$D0	;si el código ASCII de byte leído ;es < 0 entonces el carry es seteado
$008A	FIN	RTS	;el carry es cero únicamente ;cuando se lee un dígito

Dirección $0061-$0066 (97-102):
El acumulador de punto flotante número 1 (floating point accumulator



#1) es lo más importante en la ejecución de operaciones matemáticas. Este se utiliza para la conversión de números enteros a punto flotante, string a números de puntos flotante y viceversa.
El resultado de la más reciente evaluación es almacenado en esta dirección. Esta también se conoce con el nombre de FAC1.
El formato interno de almacenamiento no es fácil de entender (ni de explicar). Afortunadamente el intérprete basic dispone de varias rutinas que manipulan y convierten números. Además éstas pueden ser utilizadas por el usuario.
El FAC1 se divide a su vez en las siguientes direcciones:
$0061 (97): Esta dirección representa el exponente de FAC1 en potencia de dos. Un exponente de 128 representa el valor 0, 129 representa 2 elevado a la 0; es decir 1, 130 representa el valor 2 elevado a la 1; es decir 2, 131 es 2 al cuadrado, 132 es dos al cubo, y así sucesivamente.
$0062-$0065 (98-101): Estas direcciones representan la mantisa de FAC1. El rango de ésta está comprendido entre 1 y 1.99999... El primer bit es utilizado como signo mientras que los restantes 31 representan el valor.
Los primeros dos bytes, correspondientes a las direcciones $62 y $63, contendrán el resultado de la conversión de un número en punto flotante a entero.
$0066 (102): Esta dirección representa el signo de FAC1. Un valor de cero indica que el número es positivo, mientras que un valor de $FF (255) indicará que el número es negativo.
Con esta dirección culminamos con la descripción de FAC1.

Dirección $0067 (103):

Esta dirección es utilizada por la rutina que evalúa una expresión matemática. Indica el número de subexpresiones que deben realizarse antes de completar toda la evaluación.

Dirección $0068 (104):

Esta dirección contiene un byte denominado "overflow". Este se utiliza como paso intermedio para convertir un número entero o string a punto flotante.

Dirección $0069-$006E (105-110):

Estas direcciones representan al segundo acumulador de punto flotante o, comúnmente llamado, FAC2. Este se utiliza conjuntamente con FAC1 para realizar suma, productos, etc. El formato de éste es igual que FAC1. Es decir que está formado por:
$0069 (105): El exponente
$006A-$006D (106-109): La mantisa
$006E (110): El signo
De esta manera culminamos con la descripción de FAC2.

Dirección $006F (111):

Se utiliza para indicar al intérprete basic si los signos de FAC1 y FAC2 son iguales o distintos. Un 0 indica que los signos son iguales mientras que $FF (255) indica que los signos son distintos.

Dirección $0070 (112):

Esta dirección se utiliza en pasos intermedios de operaciones matemáticas y para redondear el cálculo final.

Dirección $0071-$0072 (113-114):

Estas direcciones apuntan a la dirección de una tabla de valores temporal dentro de la RAM libre para la evaluación de fórmulas. También se utiliza para el cálculo de arrays.

Dirección $0073-$008A (115-138):

En estas direcciones se encuentra una de las subrutinas más importantes del intérprete basic. Ella se encarga de tomar un byte del texto basic y ponerlo dentro del acumulador.

UN RATON EN LA C-64

Este programa desarrollado para la Drean Commodore 64 nos permite mover el cursor por toda la pantalla utilizando un joystick conectado en la Port número 2.



Hace unos años atrás la compañía Apple, que desarrolla computadoras personales, diseñó un periférico de entrada.
Consistía en una "bola" de libre movimiento juntamente con dos botones. Todo el conjunto se encontraba dentro de una pequeña "cajita".
El objetivo de este accesorio era que el usuario pudiera manipular el texto sin necesidad de tipear o seleccionar opciones que debían ser ingresadas por teclado. Este movimiento se lograba accionando la bola hacia la posición deseada, tomando y liberando texto a través de los botones respectivos.
Debido a que este dispositivo tenía la apariencia de un ratón a cuerda se lo bautizó con el nombre de "mouse" (ratón).
Se puede comparar con el comando utilizado para video juegos denominado "spinball".
En contraposición con el joystick que permite el movimiento en sólo ocho direcciones, el "spinball" puede mover cualquier figura a través de toda la pantalla.
No hace falta decir que aquel periférico desarrollado por Apple se sigue utilizando en nuestros días y por más computadoras. Tanto la Commodore Amiga como la PC-10 (también de Commodore) utilizan el "mouse".
Nosotros, para no quedarnos en el camino, queremos ofrecerles un programa utilitario para la Drean Commodore 64 que se parece un poco al descripto anteriormente. Nos permite mover el cursor a través del joystick, el cual debe estar conectado en la Port 2. Los movimientos son exactamente los mismos que se logran con las teclas del cursor. Es decir, sólo se permiten movimientos hacia arriba, abajo, derecha e izquierda.
El botón de disparo simula la tecla return. Sólo se puede utilizar al programar cuando se realiza algún proceso de depuración.
Se puede utilizar en modo ejecución si en una determinada línea se realiza un GET.

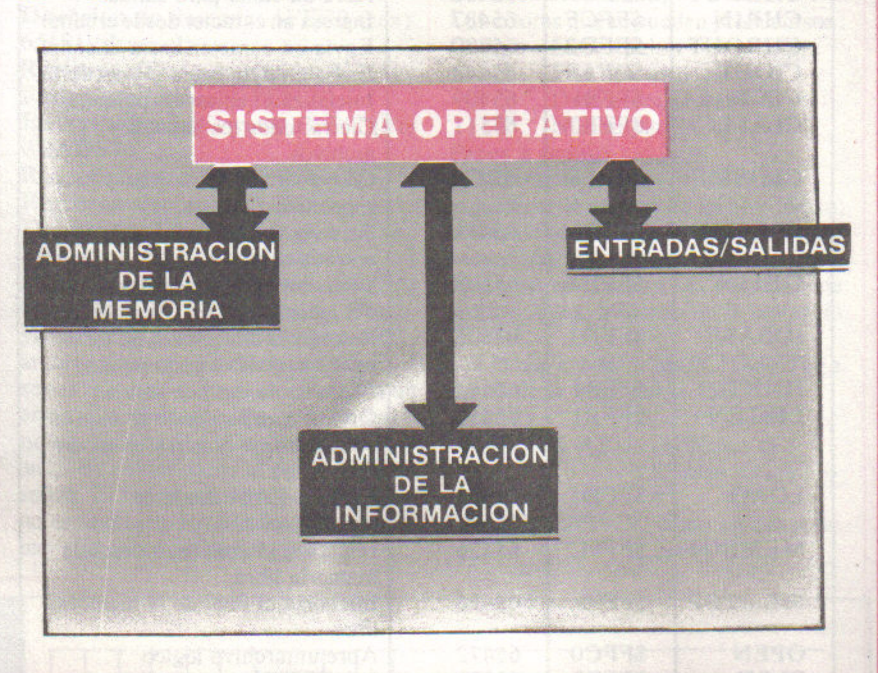
```
5   REM UN RATON EN LA C-64
10  S=49152:REM DIRECCION DE INICIO
20  FORA=STOS+84:READB:POKEA,B:C=C+B:NEXT
30  IFC<>8486THENPRINT"ERROR EN DATAS.VERIFIQUE
    LOS VALORES
    INGRESADOS.":STOP
40  POKES+7,S/256:POKES+2,S-256*PEEK(S+7)+13
50  PRINT"'RATON' ACTIVADO":SYSS
60  DATA120,169, 13,141, 20,  3,169,192,141, 21,  3
70  DATA 88, 96,230,  2,165,  2, 41,  3,208, 61,173
80  DATA  0,220, 73,255,168, 41,  1,240,  2,208, 28
90  DATA152, 41,  2,240,  2,208, 24,152, 41,  4,240
100 DATA  2,208, 20,152, 41,  8,240,  2,208, 16,152
110 DATA 41, 16,240, 23,208, 12,169,145, 44,169, 17
120 DATA 44,169,157, 44,169, 29, 44,169, 13,160,  1
130 DATA132,198,141,119,  2, 76, 49,234
```

Cómo trabaja

El programa está escrito íntegramente en lenguaje máquina ocupando poco espacio de memoria. Esta se carga a partir de la dirección decimal 49152 ($C000) accionándose a través de SYS49152.
Sin embargo es posible reubicarlo con sólo modificar la variable S (línea 10) con la nueva dirección.
Si se cambia la dirección inicial, el llamado ya no se efectúa más como antes, sino que se debe ingresar SYS S.
Para desactivarlo se debe oprimir RUN/RESTORE simultáneamente.
Su funcionamiento se logra modificando el vector que apunta a la rutina de IRQ ($EA31) la cual se encarga, entre otras cosas, de barrer el teclado 60 veces por segundo. Antes de hacer esto nos fijamos si en el port 2 se produce algún movimiento. De ser así éste se pone en el buffer de teclado y se salta a la rutina IRQ.

LAS SUBRUTINAS DEL DREAN COMMODORE 64

Comenzamos en este número a describir los subprogramas que utiliza el sistema operativo de la Drean Commodore 64. Veremos que es posible emplearlos para nuestras aplicaciones.



Describiremos las subrutinas que utiliza el KERNAL de la Drean Commodore 64. Este es el nombre del sistema operativo de la C-64.
Todas las entradas, salidas y la administración de memoria son controladas por el KERNAL. Para ello utiliza una serie de subrutinas o subprogramas cuyas funciones son bien definidas. Cada una de ellas se encarga de realizar una determinada tarea como ser sacar un caracter por pantalla, abrir un archivo, tomar un caracter de un periférico, etc.
La descripción que efectuaremos se refiere, primeramente, a la función que realiza, la dirección en donde se halla, los parámetros que ella requiere, los posibles errores en la ejecución de ésta y los registros que se involucran en la operación de la subrutina.
Saber todos estos datos nos permitirá utilizar estas rutinas. Además, al estar éstas escritas integramente en código máquina, su velocidad de ejecución es muy alta.
Antes de llamar a alguno de los subprogramas del KERNAL debemos realizar las tareas que sean necesarias. Es decir que si para ejecutar una determinada rutina debemos antes ejecutar otra, esto es exactamente lo que debemos hacer. De otra manera se producirán errores no deseados. Luego de cumplir con los requisitos, podemos acceder a ellas utilizando la instrucción JSR. Cada una de éstas está estructurada como subrutinas. Esto implica que al finalizar la ejecución, el control del programa retornará a la instrucción que sigue al JSR. En analogia con el lenguaje basic, este hecho lo podemos comparar con las sentencias GOSUB-RETURN.

Como antes mencionamos, algunas rutinas informan si se produce algún error durante la operación de éstas. Esto se realiza poniendo el código de error en el acumulador (esta operación la realiza la misma rutina). Así podemos saber qué fue lo que ocurrió para poder así tomar las medidas necesarias.

Resumiendo, las operaciones para utilizar una rutina son:
1) Setear los parámetros necesarios
2) Llamar la rutina
3) Tomar, si lo hay, el error

Antes de comenzar a describirlas, pongámonos de acuerdo en el significado de las siguientes palabras:

Nombre de función: Se refiere al nombre de la rutina.

Dirección de llamada: Es la dirección donde se encuentra.

Registros de comunicación: Son el/los registro/s que se utilizan para pasar los parámetros desde y hacia la rutina.

Rutina preliminar: Aquí se describen las rutinas que se deben ejecutar antes de llamar a la actual.

Error: Si al retornar de una rutina detectamos el flag de carry seteado (puesto a "1"), esto indicará que se produjo un error en la operación. El acumulador contendrá el número de error.

Requerimientos de stack: Indica la cantidad de bytes del stack que requiere la rutina.

Registros afectados: Aquí se listan los registros afectados por la rutina.

Descripción: Una pequeña descripción de la función de la rutina.

Pasos a seguir: Aquí se describe cómo se debe efectuar el procedimiento de llamada.

Los registros se mencionarán de la siguiente manera:
A: Acumulador
X: Registro X
Y: Registro Y

Tabla 1

Nombre	Dirección Hex	Decimal	Función
ACPTR	$FFA5	65445	Ingresa un byte desde la port serie
CHKIN	$FFC6	65478	Abre un canal para entrada
CHKOUT	$FFC9	65481	Abre un canal para salida
CHRIN	$FFCF	65487	Ingresa un caracter desde el canal
CHROUT	$FFD2	65490	Envía un caracter hacia el canal
CIOUT	$FFA8	65448	Envía un byte hacia el port serie
CINT	$FF81	65409	Inicializa el editor de pantalla
CLALL	$FFE7	65511	Cierra todos los canales y archivos
CLOSE	$FFC3	65475	Cierra un archivo lógico especifico
CLRCHN	$FFCC	65448	Reinicializa los canales de entrada-salida
GETIN	$FFE4	65508	Toma un caracter desde la cola del teclado (buffer del teclado)
IOBASE	$FFF3	65523	Retorna la dirección base de los periféricos de entrada-salida
IOINT	$FF84	65412	Inicializa la entrada-salida
LISTEN	$FFB1	65457	Ordena a un periférico situado en el bus serie a ponerse en modo "escucha"
LOAD	$FFD5	65493	Carga a RAM desde un dispositivo
MEMBOT	$FF9C	65436	Inicializa el comienzo desde la memoria libre
MEMTOP	$FF99	65433	Inicializa el final de la memoria libre
OPEN	$FFC0	65472	Abre un archivo lógico
PLOT	$FFF0	65520	Lee o fija la posición del cursor
RAMTAS	$FF87	65415	Inicializa RAM
RDTIM	$FFDE	65502	Lee el reloj de tiempo real
READST	$FFB7	65463	Lee la palabra de estado de entrada-salida
RESTOR	$FF8A	65418	Reestablece los vectores de entrada-salida
SAVE	$FFD8	65496	Almacena RAM en el dispositivo actual
SCNKEY	$FF9F	65439	Barre el teclado
SGREEN	$FFED	65517	Determina la organización de la pantalla
SECOND	$FF93	65427	Envía un comando a un periférico
SETLFS	$FFBA	65466	Inicializa un archivo lógico
SETMSG	$FF90	65424	Controla los mensajes del KERNAL
SETNAM	$FFBD	65469	Envía el nombre del archivo
SETTIM	$FFDB	65499	Setea el reloj de tiempo real
SETTMO	$FFA2	65442	Setea temporización
STOP	$FFE1	65505	Examina la tecla de STOP
TALK	$FFB4	65460	Setea un periférico para el envio de datos
TKSA	$FF96	65430	Envía un comando a un periférico luego de transmisión
UDTIM$	$FFEA	65514	Incrementa el reloj de tiempo real
UNLSN	$FFAE	65454	Ordena a los periféricos cese de recepción de datos
UNTLK	$FFAB	65451	Ordena a los periféricos cese de envios de datos
VECTOR	$FF8D	65421	Lee y setea los vectores de entrada-salida

Para comenzar les ofrecemos la tabla 1, la cual realiza una primera descripción de las rutinas:
Pasemos, pues, a describir más en detalle cada una de las rutinas:

Nombre de función: ACPTR

Propósito: Toma un dato desde el bus serie
Dirección de llamada: $FFA5 (hex) 65445 (decimal)
Registros de comunicación: A
Rutina preliminar: TALK, TKSA
Error: Ver READST
Requerimientos de stack: 13
Registros afectados: A, X
Descripción: Esta rutina se utiliza cuando queremos tomar información desde un dispositivo sobre el bus serie, como el disk drive. El dato tomado es colocado en el acumulador. Antes de utilizar esta rutina debemos primeramente llamar a la rutina TALK la cual ordena al dispositivo conectado en bus serie el envío de datos hacia la C-64. Si, además, este periférico necesita un comando secundario, éste se debe enviar a través de la rutina TKSA (también debemos llamarla antes de ejecutar la rutina ACPTR). Los errores son seteados en la palabra de estado. Para leerlos se utiliza la rutina READST.
Pasos a seguir:
1) Preparar el dispositivo seleccionado sobre el bus serie para que envíe datos hacia la C-64. Para ello utilizar las rutinas TALK y TKSA.
2) Llamar a la rutina utilizando la instrucción JSR.
3) Almacenar o utilizar el dato recibido.
Ejemplo:
(Primero llamamos a la rutina TALK que más adelante veremos)

```
0A00 JSR $FFA5 ; Tomamos un byte
0A03 STA $C000 ; y lo almacenamos
                 en la $C000
```

Nombre de función: CHKIN

Propósito: Realiza la apertura de un canal específico como entrada.
Dirección de llamada: $FFC6 (hex) 65478 (decimal)
Registros de comunicación: X
Rutina preliminar: OPEN
Error: Códigos de error 3, 5 y 6
Requerimientos de stack: No utiliza
Registros afectados: A, X
Descripción: A través de esta rutina podemos definir un determinado archivo lógico, que haya sido abierto a través de OPEN, como entrada. Desde ya el dispositivo que contenga a ese archivo debe ser un dispositivo de entrada. De otra manera se producirá un error y se finalizará la ejecución de la rutina. Si queremos tomar datos desde

otro periférico, que no es el teclado, esta rutina se utiliza antes de CHRIN y de GETIN. En cambio si utilizamos el teclado como ingreso de datos no necesitamos llamar a esta rutina ni a OPEN.
Cuando la CHKIN es utilizada con un dispositivo sobre el bus serie, ésta envía automáticamente el número de archivo lógico y, si no fue especificado en la rutina OPEN, el comando (dirección secundaria).
El registro X se utiliza para determinar el número de archivo lógico involucrado.
Como antes mencionamos, los errores se detectarán si el flag de carry del registro de estado está seteado (puesto a "1"). De ser así, el acumulador contendrá el código de error. Para esta rutina éstos pueden ser:

Código	Descripción
3	File not open
5	Device not present
6	Not input file

Pasos a seguir:
1) Abrir el archivo correspondiente.
2) Cargar en el registro X el número del archivo lógico que se utilizará.
3) Llamar a la rutina utilizando JSR.

Ejemplo:
```
0A00 LDX #$04 ; preparamos para
                entrada
0A02 JSR $FFC6 ; el archivo nro 4
```

Nombre de función: CHKOUT

Propósito: Realiza la apertura de un canal específico como salida.
Dirección de llamada: $FFC9 (hex) 65481 (decimal)
Registros de comunicación: X
Rutina preliminar: OPEN
Error: Códigos de error 0, 3, 5, 7 (Ver READST)
Requerimientos de stack: 4
Registros afectados: A, X
Descripción: A través de esta rutina podemos definir un determinado archivo lógico, que haya sido abierto a través de OPEN, como salida. Desde ya el dispositivo que contenga a ese archivo debe ser un dispositivo de salida. De otra manera se producirá un error y se finalizará la ejecución de la rutina. Esta rutina debe ser llamada antes de enviar datos al periférico de salida. El único caso en que esta rutina no es necesaria es cuando utilizamos la pantalla como periférico de salida.
Aquí también se utiliza el registro X para determinar el número de archivo lógico que será definido como canal de salida.
Cuando se utiliza un periférico conectado al bus serie, esta rutina envía la dirección de LISTEN especificada por la rutina OPEN (y, si la hubo, la dirección secundaria).
Los errores que se pueden producir son:

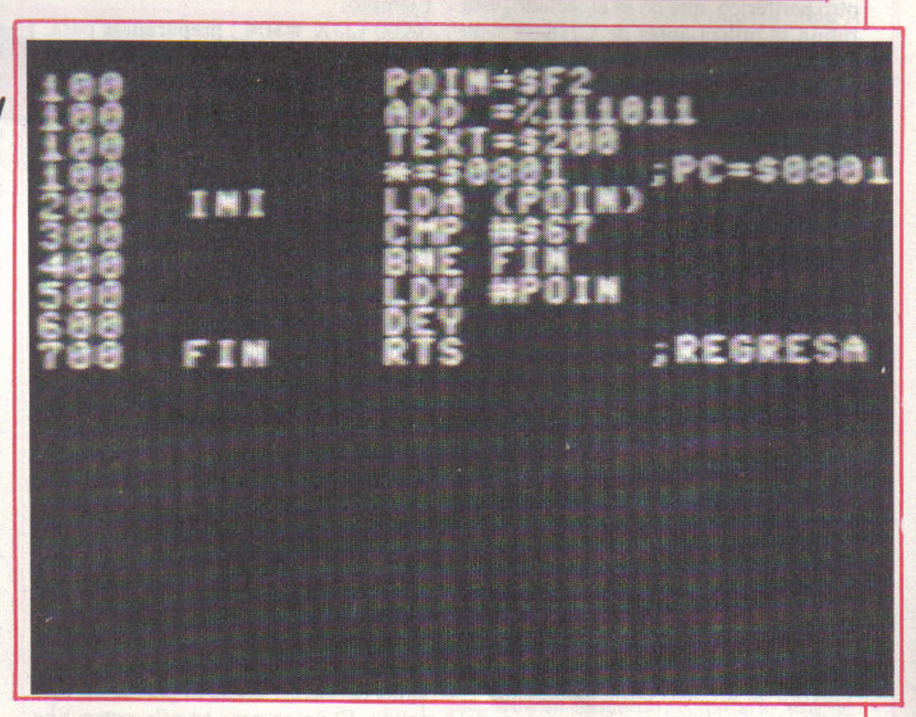
Código	Descripción
3	File not open
5	Device not present
7	Not output file

Pasos a seguir:
1) Utilizar la rutina OPEN para especificar el número de archivo lógico, la dirección LISTEN y la dirección secundaria (si se necesita).
2) Cargar en el registro X el número de archivo lógico utilizado en la sentencia OPEN.
3) Llamar a esta rutina (CHKOUT) a través de la instrucción JSR.

Ejemplo:
```
0A00 LDX #$04 ; definimos el
              ; archivo lógico nro
                4
0A02 JSR $FFC9 ; como un canal de
                 salida
```

SET DEL MICRO 6510

Continuamos describiendo el set de instrucciones del microprocesador perteneciente a la C-64



BEQ Bifurca si el resultado es cero

Operación: Salta si Z=1

N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BEQ Oper	F0	2	2

BIT Chequea bits entre memoria y acumulador

Operación: $A \wedge M, M_7 \rightarrow N, M_6 \rightarrow V$

N Z C I D V
M_7 ✓ / / / M_6

Los bits 6 y 7 son transferidos al registro de estado. Si el resultado de $A \wedge M$ es cero entonces Z=1, por lo contrario Z=0.

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Página Cero	BIT Oper	24	2	3
Absoluto	BIT Oper	2C	3	4

BMI Bifurca si el resultado es negativo

Operación: Salta si N=1

N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BMI Oper	30	2	2

BNE Bifurca si el resultado no es cero

Operación: Salta si Z=0

N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BNE Oper	D0	2	2

BPL Bifurca si el resultado es positivo

Operación: Salta si N=0

N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BPL Oper	10	2	2

BRK Pausa forzada

Operación: Interrupción forzada PC↓ P↓

N Z C I D V
/ / / 1 / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	BRK	00	1	7

BVC Bifurca si no hay overflow

Operación: Salta si V=0 N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BVC Oper	50	2	2

BVS Bifurca si hay overflow

Operación: Salta si V=1 N Z C I D V
/ / / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Relativo	BVS Oper	70	2	2

CLC Borra señalizador de transporte

Operación: 0→C N Z C I D V
/ / 0 / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	CLC	18	1	2

CLD Borra modalidad decimal

Operación: 0→D N Z C I D V
/ / / / 0 /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	CLD	D8	1	2

CLI Borra Bit desactivado de interrupción. Permite interrupción

Operación: 0→I N Z C I D V
/ / / 0 / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	CLI	58	1	2

CLV Borra señalizador de desbordamiento

Operación: 0→V N Z C I D V
/ / / / / 0

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	CLV	B8	1	2

CMP Compara memoria y acumulador

Operación: (A)−(M) N Z C I D V
✓ ✓ ✓ / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediato	CMP #Oper	C9	2	2
Página cero	CMP Oper	C5	2	3
Página cero,X	CMP Oper,X	D5	2	4
Absoluta	CMP Oper	CD	3	4
Absoluta,X	CMP Oper,X	DD	3	4
Absoluta,Y	CMP Oper,Y	D9	3	4
(Indirecta,X)	CMP (Oper,X)	C1	2	6
(Indirecta),Y	CMP (Oper),Y	D1	2	5

CPX Compara memoria y registro X

Operación: (X)−(M) N Z C I D V
✓ ✓ ✓ / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediata	CPX #Oper	E0	2	2
Página cero	CPX Oper	E4	2	3
Absoluto	CPX Oper	EC	3	4

CPY Compara memoria y registro Y

Operación: (Y)−(M) N Z C I D V
✓ ✓ ✓ / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediata	CPY #Oper	C0	2	2
Página cero	CPY Oper	C4	2	3
Absoluto	CPY Oper	CC	3	4

DEC Decrecimiento de memoria en uno

Operación: (M)−1→M N Z C I D V
✓ ✓ / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Página cero	DEC Oper	C6	2	5
Página cero,X	DEC Oper,X	D6	2	6
Absoluta	DEC Oper	CE	3	6
Absoluta,X	DEC Oper,X	DE	3	7

DEX Decrecimiento del registro X en uno

Operación: (X)−1→X N Z C I D V
✓ ✓ / / / /

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	DEX	CA	1	2

DEY Decrecimiento del registro Y en uno

Operación: (Y)−1—>Y N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	DEY	88	1	2

EOR OR Exclusiva entre acumulador y memoria

Operación: (A)⊻(M) —>A N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediato	EOR #Oper	49	2	2
Página cero	EOR Oper	45	2	3
Página cero,X	EOR Oper,X	55	2	4
Absoluta	EOR Oper	4D	3	4
Absoluta,X	EOR Oper,X	5D	3	4
Absoluta,Y	EOR Oper,Y	59	3	4
(Indirecta,X)	EOR (Oper,X)	41	2	6
(Indirecta),Y	EOR (Oper),Y	51	2	5

INC Incremento de memoria en uno

Operación: (M)+1—> M N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Página cero	INC Oper	E6	2	5
Página cero,X	INC Oper,X	F6	2	6
Absoluta	INC Oper	EE	3	6
Absoluta,X	INC Oper,X	FE	3	7

INX Incremento del registro X en uno

Operación: (X)−1—> X N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	INX	E8	1	2

INY Incremento del registro Y en uno

Operación: (Y)+1 —> Y N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Implícito	INY	C8	1	2

JMP Salto a una nueva dirección

Operación: (PC+1)—>PCL N Z C I D V
(PC+2)—>PCH //////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Absoluto	JMP Oper	4C	3	3
Indirecto	JMP (Oper)	6C	3	5

JSR Salta a una nueva dirección conservando la dirección de retorno

Operación: PC+2 ↓, (PC+1)—> PCL N Z C I D V
(PC+2)—> PCH //////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Absoluto	JSR Oper	20	3	6

LDA Carga acumulador con memoria

Operación: (M)—>A N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediato	LDA #Oper	A9	2	2
Página cero	LDA Oper	A5	2	3
Página cero,X	LDA Oper,X	B5	2	4
Absoluta	LDA Oper	AD	3	4
Absoluta,X	LDA Oper,X	BD	3	4
Absoluta,Y	LDA Oper,Y	B9	3	4
(Indirecta,X)	LDA (Oper,X)	A1	2	6
(Indirecta),Y	LDA (Oper),Y	B1	2	5

LDX Carga el registro X con memoria

Operación: (M) —>X N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediato	LDX #Oper	A2	2	2
Página cero	LDX Oper	A6	2	3
Página cero,Y	LDX Oper,Y	B6	2	4
Absoluta	LDX Oper	AE	3	4
Absoluta,Y	LDX Oper,Y	BE	3	4

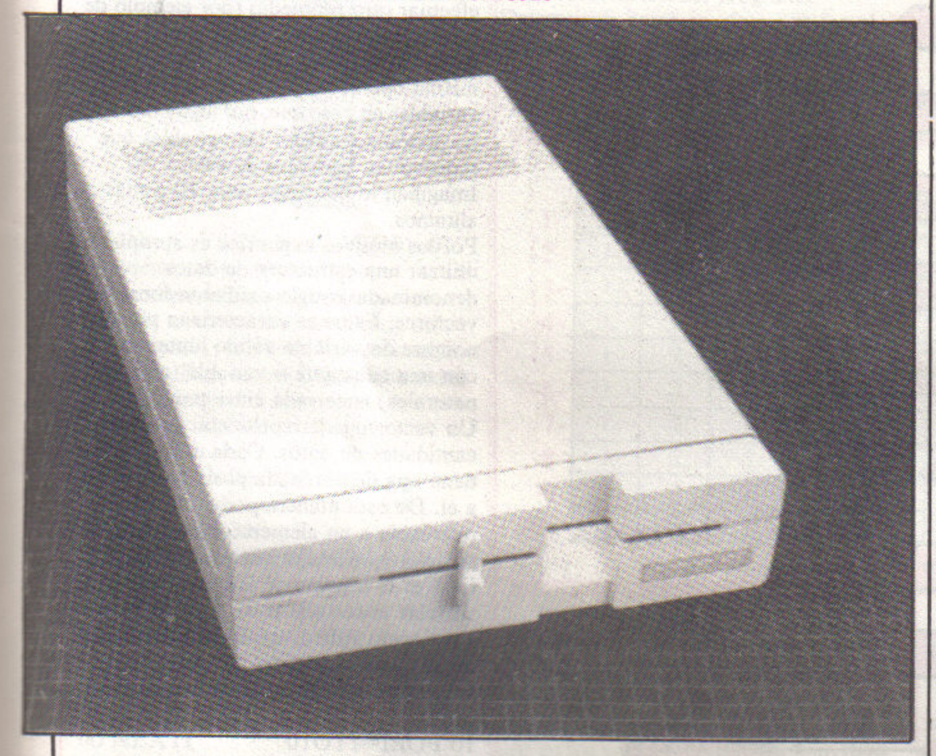
LDY Carga el registro Y con memoria

Operación: (M) —>Y N Z C I D V
√√////

modo de direccionamiento	mnemotécnico	CO	Nro. de bytes	Nro. de ciclos
Inmediato	LDY #Oper	A0	2	2
Página cero	LDY Oper	A4	2	3
Página cero,X	LDY Oper,X	B4	2	4
Absoluta	LDY Oper	AC	3	4
Absoluta,X	LDY Oper,X	BC	3	4

MANEJO DEL DRIVE 1571

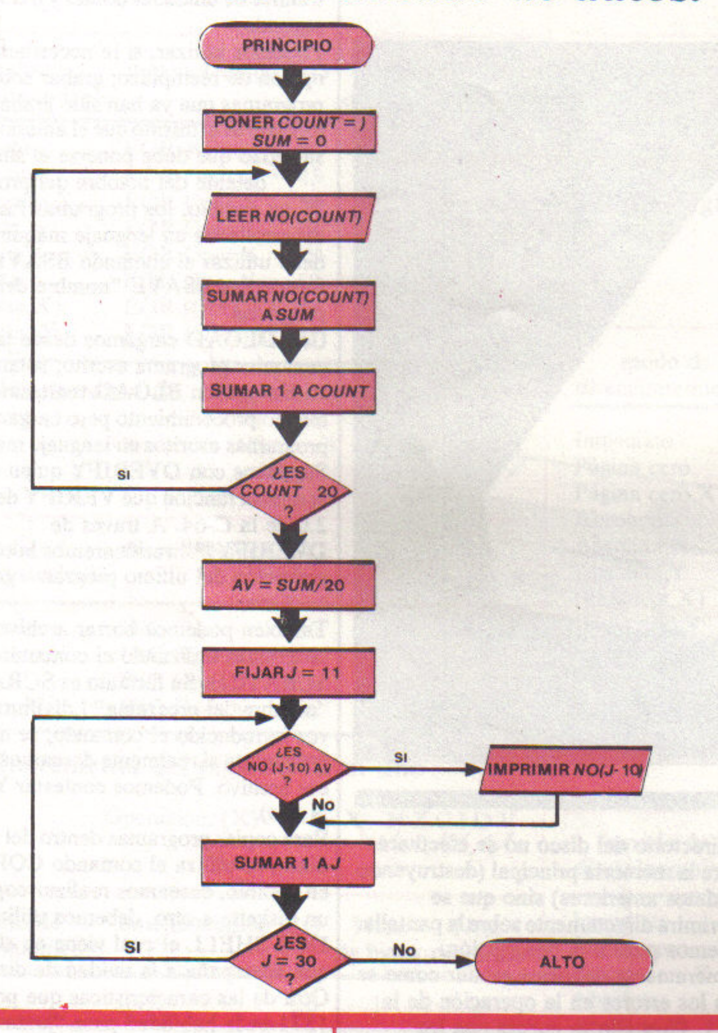
Es uno de los más recientes periféricos lanzados juntamente con la C-128, una de las más potentes "home computers" del mercado mundial.



Como se sabe, esta computadora puede trabajar en tres modos: 64, en donde se comporta exactamente como una C-64; modo 128, donde el basic residente es el 7.0 y, finalmente, CP/M.
La 1571 se adapta perfectamente a cada uno de estos modos. En el primero, ella se comporta como una 1541 cien por cien. En el segundo se adapta a los requerimientos del 7.0, aumenta su velocidad de transferencia de información y, además, aumenta su capacidad de almacenaje. En el último, la velocidad aumenta de 8 a 10 veces y su capacidad se duplica respecto a la de la 1541.
El Basic 7.0 incluye una larga lista de comandos orientados al uso del drive 1571. Esos comandos posibilitan trabajar directamente con ella sin interferir con los demás elementos que constituyen el equipo. Con esto queremos decir que, por ejemplo, cargar el directorio del disco no se efectuará sobre la memoria principal (destruyendo los datos anteriores) sino que se imprimirá directamente sobre la pantalla.
Pasemos pues a la descripción.
Primeramente debemos relatar como se leen los errores en la operación de la 1571. Para ello se cuenta con las variables DS y DS$ que contienen el número y la descripción del error. Es decir que si la luz verde del drive comienza a destellar, podemos visualizar rápidamente qué fue lo que sucedió tipeando PRINT DS,DS$.
Con respecto a la 1541, esta tarea se lleva a cabo haciendo OPEN,8,15:INPUT#15,A,B$,C,D:PRINTA,B$,C,D;CLOSE15.
Además, esta línea, no se puede ejecutar en modo directo.
Para grabar los programas se utiliza el comando DSAVE. Este se utiliza para almacenar los programas escritos en basic solamente. Su formato es DSAVE "nombre del programa" [,nrol][,dis] en donde nro y dis son opcionales y representan número de drive (0 ó 1 para el primero) en caso de tratarse de unidades dobles y 8 ó 9 para el segundo.
Podemos utilizar, si la necesitamos, la opción de reemplazo; grabar sobre programas que ya han sido grabados. El formato es el mismo que el anterior con la salvedad que debe ponerse el símbolo " " delante del nombre del programa.
Si, en cambio, los programas fueron desarrollados en lenguaje máquina, se debe utilizar el comando BSAVE cuyo formato es BSAVE "nombre del programa"
Con DLOAD cargamos desde la 1571 cualquier programa escrito, solamente, en basic. Con BLOAD realizamos el mismo procedimiento pero cargamos los programas escritos en lenguaje máquina.
Seguimos con DVERIFY quien realiza la misma función que VERIFY del basic 2.0 de la C-64. A través de DVERIFY"*" verificaremos la correcta grabación del último programa grabado en disco.
También podemos borrar archivos y programas utilizando el comando SCRATCH. Su formato es SCRATCH "nombre del programa" [,dis][nro]. Una vez introducido el comando, se nos preguntará si realmente deseamos borrar ese archivo. Podemos contestar Y (si) o N (no).
Para copiar programas dentro del mismo disco, se utiliza el comando COPY. Si, en cambio, deseamos realizar copias de un diskette a otro, debemos utilizar el DOS SHELL el cual viene en el disco que acompaña a la unidad de disco.
Otra de las características que posee la 1571 es la habilidad para "juntar" dos archivos secuenciales. Esto se logra a través del comando CONCAT.
Si, en cambio, queremos agregar datos a un archivo o un programa, esto se logra utilizando el comando APPEND.
Se incluyen, también, los comandos para cambiar el nombre a un determinado programa, inicializar el disco y organizar el disco (RENAME, INICIALITE y COLLECT).
Para inicializar el disco se utiliza el comando HEADER.
El 7.0 posee el comando BOOT, el cual carga un programa en lenguaje máquina e inicia su ejecución.

VECTORES Y MATRICES

Les comentamos cómo sacarle el jugo al almacenamiento ordenado de datos.



Vectores

Una de las formas de efectuar un ingreso de datos por teclado o desde algún medio de almacenamiento externo, es utilizando una lista de entrada con los nombres de las variables respectivas. Es decir que si debemos ingresar determinados valores desde un dispositivo dado, relacionado con un número lógico de archivo, el procedimiento sería:

INPUT#Nro,A,B,C,D,...

donde Nro es el número de archivo seleccionado. Desde ya este INPUT depende de cada computadora. El nuestro corresponde al caso de la C-64. Un ejemplo en concreto es la lectura del canal de error del disk drive 1541:

```
INPUT#15,A,B$,C,D
```

Si hubiésemos querido ingresar información a través del teclado, tendríamos que cambiar el formato del comando; es decir sin el símbolo de numeral. Pero a medida que vamos aumentando la cantidad de variables, el manejo de ellas resulta bastante complicado. Supongan que debemos efectuar el promedio de tres valores, esto sería:

```
10 INPUTA,B,C
20 PR=(A+B+C)/3
30 PRINTPR
40 END
```

Ahora si el promedio a realizar es de 10 valores, tendríamos que cambiar la línea 10 y 20 por:

```
10 INPUTA,B,C,D,E,F,G,H,I,J
20 PR=(A+B+C+D+E+F+G+H+I+J)/10
```

Supongan, además, que debemos efectuar otro promedio (por ejemplo de otro alumno) y determinar cuál fue el mayor. Necesitamos para ello repetir la misma operación y utilizar otras variables en caso que, por algún motivo, no podramos utilizar las variables que representan las notas de éste.

Imaginen lo que sería para 10 ó más alumnos.

Por los motivos expuestos es apropiado utilizar una estructura de datos denominada arreglo unidimensional o vectores. Estos se caracterizan por un nombre de variable válido juntamente con una constante o variable (ambas naturales) encerrada entre paréntesis. Un vector puede representar grandes cantidades de datos. Cada uno de ellos tiene una determinada posición relativa a él. De esta manera podemos hacer referencia a un elemento diciendo la posición que ocupa dentro del vector. Esta es la función de la constante o variable encerrada entre paréntesis: sirve como índice; apunta a la dirección del elemento buscado. Un ejemplo de asignación de elementos a un vector sería:

```
10 FORI=1TO10
20 V(I)=I
30 NEXTI
40 END
```

Así el primer elemento del vector V es el 1, el segundo elemento es el 2, así hasta llegar al décimo (décimo elemento corresponde al 10). Podemos imaginar la representación en memoria de este hecho como:

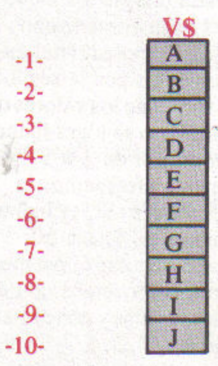
	V
-1-	1
-2-	2
-3-	3
-4-	4
-5-	5
-6-	6
-7-	7
-8-	8
-9-	9
-10-	10

V(1)=1, V(2)=2, ..., V(10)=10

De igual manera podemos almacenar strings o, simplemente, caracteres:

```
10 FORI=65TO74
20 V$(I−64)=CHR$(I)
30 NEXTI
40 END
```

Luego la distribución de los elementos quedaría:



V$(1)=A, V$(2)=B, ..., V$(10)=J

Volviendo al primer ejemplo, desarrollemos un programa que nos permita calcular el promedio de un determinado alumno ahora utilizando vectores. Supongamos que la cantidad de notas leídas es 10.

```
10 REM LEEMOS NOTAS
15 SU=0
20 INPUT"NOMBRE DEL
ALUMNO"; AL$
25 FORI=1TO10
30 PRINT"INGRESE NOTA Nro";I
35 INPUTNO(I)
40 NEXTI
45 REM EFECTUAMOS
CALCULO PROMEDIO
50 FORI=1TO10
55 SU=SU+V(I)
60 NEXTI
65 PR=SU/10
70 PRINT"EL PROMEDIO DEL
ALUMNO";AL$;"ES ";PR
75 GOTO15
```

Este programa nos serviría para hallar el promedio de diversos estudiantes. Observen que cuando finalizamos el cálculo de uno de ellos, regresamos al principio para leer un nuevo nombre con sus correspondientes notas y seteamos a cero la sumatoria (SU) anterior. Miren la línea 55 para comprobar esto. Un posible ejercicio sería, pues, calcular el promedio sin saber cuántas notas se leerán. Efectúen el programa respectivo, el cual debe leer el total de notas de cada alumno. Analicen el caso en que se ingrese cero notas, un número no entero o un número mayor a 10. De acuerdo a sus resultados contemplen estos casos en la elaboración del programa. Determinen, también, de quién fue el promedio más alto. Impriman el nombre juntamente con el promedio.

Figura 1

	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
-1-	1	2	3	4	5	6	7	8	9	10
-2-	1	2	3	4	5	6	7	8	9	10
-3-	1	2	3	4	5	6	7	8	9	10
-4-	1	2	3	4	5	6	7	8	9	10
-5-	1	2	3	4	5	6	7	8	9	10
-6-	1	2	3	4	5	6	7	8	9	10
-7-	1	2	3	4	5	6	7	8	9	10
-8-	1	2	3	4	5	6	7	8	9	10
-9-	1	2	3	4	5	6	7	8	9	10
-10-	1	2	3	4	5	6	7	8	9	10

Matrices

Así como los vectores son arreglos unidimensionales, las matrices son arreglos bidimensionales. Aquí, al igual que en los vectores, cada elemento tiene una determinada posición dentro de la matriz. La diferencia es que para acceder a éste se deben indicar dos parámetros: la fila y la columna donde se encuentra. Es decir que habrá dos variables o constantes (ambas naturales) que se utilizarán como índice juntamente con el nombre de la matriz. Es decir: V(I,J) donde I representa la fila y J la columna de un determinado elemento. Haciendo la siguiente asignación:

```
10 FORI=1TO10
20 FORJ=1TO10
30 V(I,J)=J
40 NEXTJ
50 NEXTI
60 END
```

la distribución de los elementos dentro de la matriz es según Figura 1.

V(1,1)=V(2,1)=V(3,1)=...
=V(10,1)=1
V(1,2)=V(2,2)=V(3,2)=...
=V(10,2)=2

V(1,10)=V(2,10)=V(3,10)=...
=V(10,10)=10

Para continuar ejemplificando lo dicho, regresemos al primer ejemplo dado y reformulemos y modifiquemos el problema para que podamos resolverlo utilizando ahora matrices:
Se codifican los nombres de los alumnos pertenecientes a un determinado curso usando los primeros cinco números naturales. De esta forma al alumno

ESTRUCTURA DE DATOS

ALBATE le corresponde el número 1, a BENITEZ el número 2, a CALANDRIA el 3, MARTINEZ el 4 y, finalmente, UBERTAZZI a quien le corresponde el número 5.
Además, cada alumno viene acompañado por los siguientes datos personales: su edad, nota1, nota2, nota3, nota4, nota5, nota6, nota7, año de ingreso, año de egreso. Este último item contendrá el valor 0 si el estudiante aún permaneciese en la casa de estudios o, en caso contrario, el año de su egreso.
Se pide:
1) Hallar la edad promedio del curso.
2) Hallar la nota promedio de cada alumno.
3) Imprimir aquellos alumnos que hallan ingresado en el 83.
4) Imprimir, si los hubo, los alumnos que hallan egresado.
Los datos que se tienen son según Figura 2.

Lo primero que haremos será ingresar los datos por fila, es decir por cada alumno. Luego iremos desarrollando cada bloque de programa acorde a las metas prefijadas:

Figura 2

Nro	Edad	Nota1	Nota2	Nota3	Nota4	Nota5	Nota6	Nota7	Ingre	Egre
1	19	5	7	8	7	6	7	6	83	0
2	20	8	8	8	5	4	7	7	82	83
3	21	9	8	7	4	7	8	8	82	0
4	20	9	9	9	8	8	8	8	83	0
5	20	6	6	9	8	8	7	9	82	84

```
1 REM VECTORES Y MATRICES-CRISTIAN PARODI
5 DIM V(5,11)
7 REM INGRESAMOS DATOS
10 FORI=1TO5
15 PRINT“INGRESE DATOS DEL ALUMNO”;I
20 FORJ=1TO11
25 PRINT“INGRESE ITEM”;J;“DEL ALUMNO”;I
30 INPUTV(I,J)
35 NEXTJ
40 NEXTI
45 REM HALLAMOS EDAD PROMEDIO
50 SU=0
55 FORI=1TO5
60 SU=SU+V(I,2)
65 NEXTI
70 PRINT“EDAD PROMEDIO DEL CURSO:”;SU/5
75 REM HALLAMOS NOTA PROMEDIO
80 FORI=1TO5
85 SU=0
90 FORJ=3TO9
95 SU=SU+V(I,J)
100 NEXTJ
105 PRINT“EL PROMEDIO DEL ALUMNO”;I;“ES”;SU/7
110 NEXTI
115 REM INGRESOS EN EL 83
120 F=0:FORI=1TO5
125 IFV(I,10)=83THENPRINT“EL ALUMNO”;I;“INGRESO EN EL 83”:F=1
130 NEXTI
132 IFF=0THENPRINT“NO HUBO INGRESOS EN EL 83”
135 REM EGRESOS DE ALUMNOS
140 F=0:FORI=1TO5
145 IFV(I,11)<>0THENPRINT“EL ALUMNO”;I;“EGRESO EN EL”;V(I,11):F=1
150 NEXTI
155 IFF=0THENPRINT“NO HUBO EGRESOS”
160 END
```

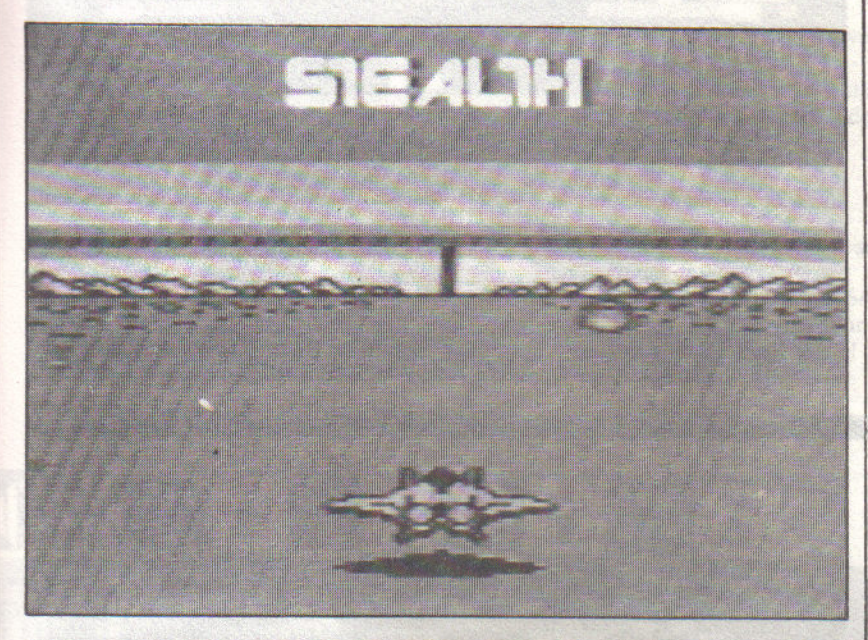
La sentencia DIM

Antes de comenzar a explicar el programa, hablemos un poco sobre la sentencia DIM. Esta se utiliza para indicarle al intérprete Basic el espacio de memoria que debe reservar para el almacenamiento de los elementos de vectores o de matrices. Esta sentencia puede omitirse si se trabaja con vectores cuyos elementos no superan los diez o si se trabaja con matrices menores o iguales a las 10 filas y 10 columnas (es decir 10x10). En estos casos se asume por default (valor preestablecido) que se trabajará con 10 elementos para los vectores y 10,10 para las matrices. Observen que en los ejemplos citados anteriormente no se utilizó la sentencia DIM por estar trabajando con cantidades de elementos válidos. Prueben ustedes aumentar a 11 los elementos del vector y analicen el mensaje de error que emitirá el intérprete.
Ahora, vayamos a la explicación del programa:

Primeramente, en la línea 5, dimensionamos la matriz en forma adecuada. Luego procedemos a ingresar los datos de cada alumno (ingresamos los datos por fila). De esta manera ingresamos el número de código del alumno, la edad, nota1, así hasta llegar hasta el año de egreso. Repetimos esta tarea para cada uno de los cinco estudiantes. Luego comenzamos a calcular la edad promedio del curso. Debemos movernos por la segunda columna e ir sumando los valores que allí se encuentran. Esto se logra fijando el J en 2 y variando el I de 1 a 5. Simultáneamente efectuamos la sumatoria respectiva (SU) y hallamos el promedio (miren las líneas 50, 55, 60, 65 y 70). Para calcular el promedio de cada alumno nos movemos con el I a través de todas las filas y con el J a través de las columnas 3, 4, 5, 6, 7, 8 y 9 que es donde se encuentran nota1, nota2, nota3,... nota7. Cada vez que imprimimos el promedio de un estudiante seteamos la sumatoria (SU) a cero y repetimos la tarea. Esta está controlada por el I (líneas 80,85,90,95,100,105,110). A continuación chequeamos si hubo un ingreso en el 83. Esto se logra observando si se encuentra el valor 83 en la columna 10 de la matriz (líneas 120,125,130). Por último comprobamos si hubo algún egreso viendo si algún elemento de la columna 11 es distinto de 0 (líneas 140,145,150).
Para culminar con la nota les dejamos los siguientes ejercicios:
En el anterior problema, encuentren cuál fue el promedio más alto y cuál fue el más bajo. Si hicieron el anterior no tendrán problemas en resolver éste. Además, calculen el promedio general del curso informando, al mismo tiempo, qué alumnos tuvieron notas inferiores a 5. Prueben efectuar un algoritmo que, dados una nota y la edad, imprima el/los alumno/s que tienen esos datos. Desde ya nuestra revista tiene sus puertas abiertas para responder a sus dudas e inquietudes.

A SUPERIOR
B MUY BUENO
C BUENO
D REGULAR
E MALO
F INACEPTABLE

STEALTH



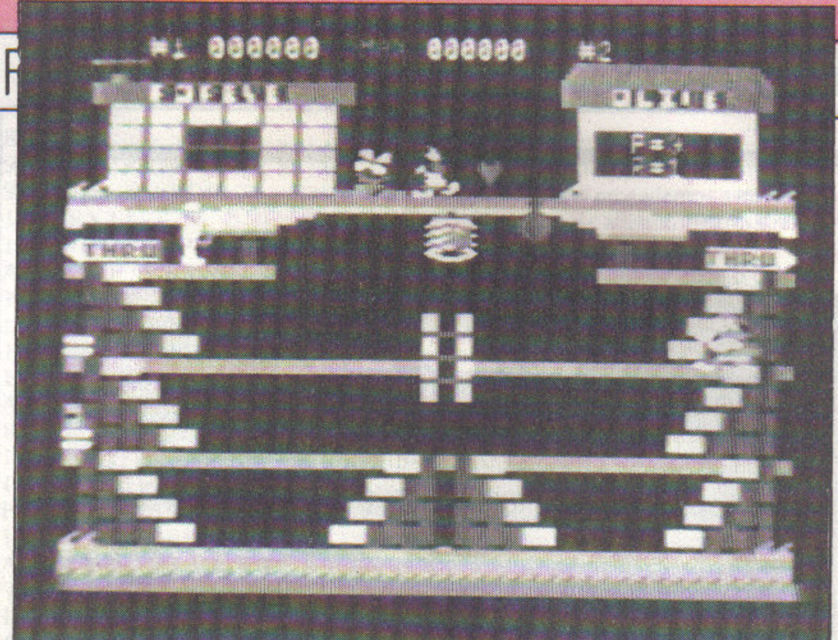
Rating total: A
Creatividad: A
Documentación: B
Valor en relación al costo: B+
Computadora: Drean Commodore 64
Editor: Peek

Imaginen la perspectiva que se observa desde un avión que vuela al ras del piso a gran velocidad y que el enemigo le dispara misiles y bombas. Si no lo pueden imaginar entonces les recomendamos jugar con STEALTH. En él verán efectos especiales de video muy interesantes.
Nuestra misión consiste en destruír la fortaleza del dictador Tehconip. Para ello contamos con tres naves. Cada una de ellas se comanda a través del joystick. Podemos ir, solamente, hacia la izquierda y hacia la derecha.
La nave se desplaza, como antes mencionamos, al ras del piso. Por ello no se permiten movimientos hacia arriba ni hacia abajo. Lo único que podemos hacer es acelerar o desacelerar moviendo la palanca de mandos hacia adelante o hacia atrás.
El enemigo cuenta con instalaciones, como radares, misiles, tanques entre otros, las cuales se encuentran dentro del perímetro por el cual transitamos y cuya finalidad es impedirnos cumplir con nuestro objetivo. Estas están instaladas a lo largo de las 10.000 millas que debemos recorrer.
Además de evitar que ellos nos destruyan, debemos cumplir con la misión antes de que nuestra energía se agote. De otra manera la nave se autodestruirá.
El juego tiene varios niveles de acción. Para destruír la fortaleza se requiere de un disparo directo, en nivel 1. Luego, en el 2, de dos disparos directos, y así sucesivamente. En la pantalla aparece en qué nivel estamos, la energía que nos resta, las naves que nos quedan, el score y el tiempo transcurrido.
Con respecto a la dificultad de los distintos niveles podemos decir que el primero es fácil de atravesar, el segundo no tanto y el tercero sería mejor que hablasen los especialistas!!!.
Al comienzo del juego se nos explica cuáles son las distintas instalaciones del enemigo, cómo se representa un volcán activo y uno "apagado", cómo sabemos si nuestros motores están en reversa o si están acelerando. Aquí seleccionamos el nivel del juego.
Deseamos resaltar el manejo excelente de gráficos, simulando perspectivas asombrosas. De los misiles del enemigo podemos decir que menos mal que se trata de un juego. Decidan ustedes, en base a esto, si vale la pena o no darse un paseo de 10.000 millas esquivando y destruyendo bases.

POPEYE

Rating total: B+
Creatividad: A
Documentación: B
Valor en relación al costo: B
Computadora: Drean Commodore 64
Editor: Peek

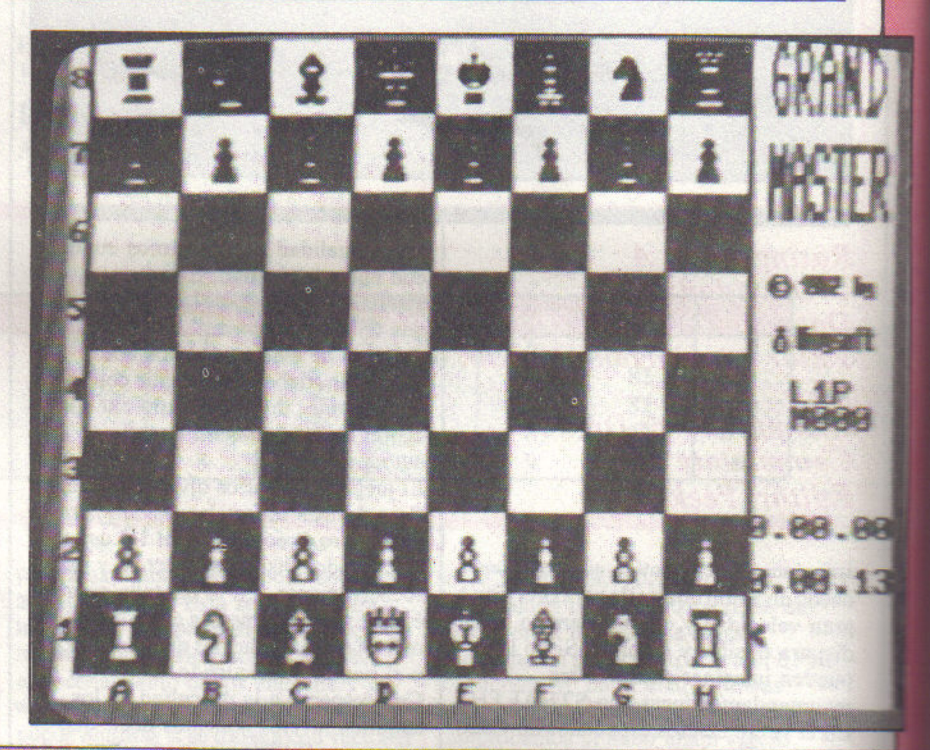
Seguramente, no hará falta describir quién fue y quién es este personaje. Tendríamos que citar, para no dejarlos a un lado, a los amigos y no tan amigos que siempre lo acompañan en sus aventuras. Ellos son la simpática Olivia y su temible enemigo Brutus.
Por supuesto no podía pasar que los fabricantes de software no rindiesen su homenaje al héroe máximo de la espinaca. Para ello diseñaron un video game. Drean Commodore lo adaptó a sus equipos para presentar su propia versión denominado POPEYE.
El juego consiste básicamente en rescatar a Olivia de las garras de Brutus. Nuestro amigo, comandado por el joystick, debe salvarla.
Los escenarios de la confrontación Popeye-Brutus son tres. El primero transcurre dentro de un edificio.
Nuestra misión consiste en recolectar una determinada cantidad de corazones enviados por la triste Olivia.
Una vez hecho esto habremos liberado a la novia de Popeye. Estos caen suavemente desde el último piso hasta la planta baja. Deberemos a toda costa atajarlos antes de que lleguen al suelo. De lo contrario terminaremos con una de las tres vidas de Popeye.
Brutus nos persigue por todo el edificio arrojándose, si puede varias botellas. Además aparecen, de vez en cuando, dos secuaces de él tratando de eliminarnos. Por suerte el barril de espinaca está a nuestro alcance. Claro que, cuando la tomemos, Brutus comenzará una rápida fuga evitando el feroz castigo de Popeye.
Una vez que hayamos juntado los corazones, pasamos al segundo escenario. Este representa la construcción de un edificio. Aquí debemos juntar notas musicales. Desde ya Brutus y sus inseparables amigos, tratarán de destruírnos.
Por suerte hay muchas escaleras y trampolines que nos ayudan a escapar. Además sigue habiendo espinaca. Lamentablemente no hay canilla libre,



sólo se permite una consumición!!! El último escenario consiste en una serie de niveles. Olivia en el más superior. Para rescatarla tenemos que juntar la palabra HELP enviada por ella. A medida que las juntamos se va formando una escalera. Una palabra HELP que tomemos incrementará la altura de ella.
Brutus seguirá impidiendo nuestra labor sumándose, aquí, un raro pajarraco cuyas intenciones son más que hostiles. Si somos tan astutos de pasar este nivel, el juego comenzará nuevamente. Pueden participar dos jugadores (sólo con joystick).
POPEYE es un excelente programa, con figuras más que parecidas a los personajes reales.

CHESS 64

Rating total: A
Creatividad: B+
Documentación: B
Valor en relación al costo: B+
Computadora: Drean Commodore 64
Editor: Peek



Este es una de las primeras versiones de ajedrez desarrollado para la C-64. Luego vinieron CHESS 7.0 y SARGON III. Todos ellos, realizados utilizando técnicas de inteligencia artificial, varian en el nivel de juego. Sólo comentaremos el CHESS 64. Para los que gustan de este juego encontrarán

un difícil e inteligente oponente.
Luego de cargarlo en la memoria de la computadora, el programa se auto ejecuta apareciendo sobre la pantalla el tablero, con sus respectivas coordenadas, con las piezas.

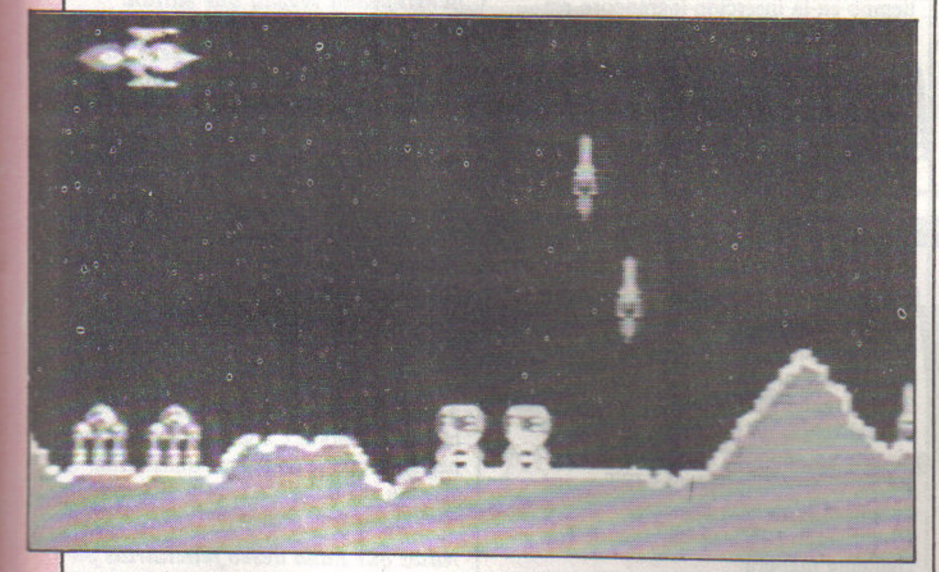
De no agradarnos el color de todo el conjunto, se nos permite cambiarlo por el deseado con sólo oprimir las respectivas teclas de función. Por ejemplo F1 cambia el color del borde, F2 el color del tablero, etc.
Debemos tener en cuenta que una vez que aparece el tablero en pantalla, CHESS 64 espera el ingreso de nuestra jugada.

Para ello dispone de dos contadores. Uno contabiliza el tiempo de demora en realizar nuestra jugada. El otro cumple la misma función pero se activa solamente cuando CHESS 64 comienza a "pensar" su jugada.

Para indicarle qué ficha vamos a mover y dónde la pondremos, debemos determinar la coordenada en donde ella se encuentra y la coordenada a donde irá.

Luego las tipeamos oprimiendo finalmente la tecla return.
Si la movida fue legal, nuestra pieza se trasladará al lugar deseado conmutándose el tiempo a CHESS 64. Si en cambio tratamos de mover, por ejemplo, un caballo como un alfil, el movimiento no se llevará a cabo siguiéndose contabilizando nuestro tiempo de demora.
CHESS 64, como antes dijimos, es un buen contrincante para los aficionados al ajedrez. Tiene, lamentablemente, una pequeña falla: no es capaz de determinar un empate. Por ejemplo, si nosotros nos quedamos solamente con nuestro rey y él con su caballo y su rey, es imposible que nos de mate con esas dos piezas. Lo que sucede en esta situación, es que seguimos escapando del ataque repitiéndose, en algún momento, estados anteriores.
De todas maneras insistimos: es un muy buen juego de ajedrez. No cometan el error que cometimos nosotros al subestimarlo: el score fue Redacción:0, CHESS 64:8!!!.

SKRAMBLE



Rating total: B—
Creatividad: B
Documentación: B
Valor en relación al costo: B
Computadora: Drean Commodore 64
Editor: Peek

El objetivo de este divertido y peligroso juego es destruír la mayor cantidad de naves, tanques de abastecimiento y edificios enemigos. Para tal fin comandamos una nave espacial, la cual dispara misiles y bombas simultáneamente. Está en continuo movimiento (siempre avanzando).
Nosotros solamente podemos mover la nave hacia arriba, hacia abajo y hacia atrás.
Para realizar la misión disponemos de tres artefactos, los cuales deben atravesar seis niveles de juego. Para ello tenemos, durante el desarrollo del juego, un pequeño indicador de niveles, el cual nos indica en cuál estamos.
El primero consiste simplemente en esquivar o destruir los proyectiles que nos disparan y, simultáneamente, destruír los tanques de combustible.
Esto no lo debemos descuidar ya que nuestra nave depende de un campo de energía, el cual se llena cada vez que destruímos estos tanques. Si así no lo hiciéramos, nuestra reserva de energía llegará a cero ocurriendo, de esta manera, nuestra autodestrucción.
A partir del segundo nivel el desarrollo del juego se va complicando.
Aquí debemos seguir bombardeando los tanques de combustible y esquivar la flota kamikaze que no dudará en estrellarse contra nosotros.
Por suerte podemos destruírlos, ya sea a través de las bombas o de los misiles.
En el tercer nivel debemos esquivar unos meteoritos y, por supuesto, seguir eliminando los famosos tanques, en caso de que no querramos perecer inmediatamente.
Los meteoritos son indestructibles. Es decir que por más que disparemos nuestras bombas y misiles ellos seguirán en su inamovible curso.
Es muy conveniente no estar en línea recta con alguno de ellos!
Antes de ingresar al cuarto nivel debemos estar totalmente seguros de que nuestros reflejos estén en óptimas condiciones. Aquí entramos dentro de la fortaleza del temible general Olam, el cual nos tiene preparada curiosas sorpresas. Debemos maniobrar con mucho cuidado sin que nuestra nave se estrelle contra el fuerte y sin que ningún proyectil enemigo nos alcance.
Siguiendo, los túneles comienzan a hacerse cada vez más empinados. En uno de los sectores debemos subir y bajar prácticamente en ángulo recto.
Si podemos atravesarlo sin chocar, habremos cumplido con nuestra misión, apareciendo en la pantalla "FELICITACIONES CAPI". Luego se reinicia el juego.
SKRAMBLE es un entretenimiento con gran variedad de figuras y diversos sonidos.

C-64

De mi mayor consideración:
Me dirijo a Uds. para solicitarles la gentileza de contestarme las siguientes consultas:
1) Poseo una Commodore 64 norteamericana. Existe alguna diferencia con la Drean Commodore en cuanto a las posiciones de memoria o alguna otra característica importante a tener en cuenta para utilizar la información que aparece en su revista?
2) He desarrollado algunos programas bastante extensos, en Basic, para contabilidad, que utilizan vectores de hasta 1.000 elementos. Noto que a medida que el vector se va "llenando" la operación se va haciendo más lenta, y en uno de los programas, en que debe imprimir con la distribución adecuada los distintos registros de un vector que habitualmente se llena hasta alrededor del elemento Nº 900, la impresión suele paralizarse por espacio de alrededor de un minuto, continuando luego con total normalidad, por lo cual el único inconveniente es la demora. ¿Hay alguna manera de evitarla o disminuirla?
3) Me interesaría encontrar algún medio para que un programa en Basic en cassette, forzosamente se ejecute automáticamente al cargarlo. Desde luego, esto puede lograrse pulsando simultáneamente SHIFT y RUN/STOP, pero el operador no está obligado a hacerlo y puede pedir LOAD y luego RUN. El quid de la cuestión está en que quiero evitar que pueda pedir un LIST luego de hacer LOAD y antes del RUN. Después ya no hay problema, porque una de las primeras instrucciones es un POKE 808,234 que inhabilita la tecla RUN/STOP, e invariablemente mis programas terminan con un SYS 64738 que destruye toda la información. Me serviría igual algún método que, aunque no provoque la ejecución automática, bloquee el LIST aún antes de hacer RUN. ¿Es esto posible Desde ya les quedo muy agradecido por su atención, y los saludo atentamente.
Dr. Daniel Alberto Cotarelo García
Capital Federal

1) No existe absolutamente ninguna diferencia entre la Commodore 64 y la Drean Commodore 64. Todos los datos y notas que suministramos son cien por cien compatibles con una y otra máquina.
2) Es aconsejable inicializar todas las variables (incluyendo vectores) antes de comenzar a realizar el programa en sí. La ventaja radica en que el intérprete Basic debe hacer un espacio en memoria cada vez que se encuentra con una variable que no haya sido inicializada. "Hacer un espacio', es una tarea muy lenta si la cantidad de variables utilizadas es grande ya que debe mover todas ellas (string, vectores, etc.) siete posiciones de memoria para poder insertar la nueva variable. De la otra forma, es decir inicializándolas a cero (por ejemplo), el intérprete Basic no pierde tiempo en la inserción lográndose así mayor velocidad de ejecución. De todas maneras el aumento de la velocidad de ejecución será aún más notable si todas las operaciones de inserción de un elemento y llenado del vector se realizarán en una subrutina escrita en lenguaje máquina. Para ello se necesitan tener conocimientos de almacenamiento interno de los números en la C-64, áreas de memorias específicas y, por supuesto, manejar muy bien la programación en lenguaje máquina o assembler del micro 6510. Para no frustrar tus inquietudes prueba con el método de inicialización.
3) Una manera de evitar lo que tu describes es realizando un programa el cual, al ejecutarse, bloquee las teclas de RUN/STOP, bloquee el LIST y que cargue el programa principal.
El comando LIST se bloquea de la siguiente manera:
POKE 774,148 : POKE 775,227
Si luego de ejecutar esta sentencia ingresas el comando LIST, se producirá el mismo efecto de SYS 64783. Otra manera es:
POKE 774,0
Aquí sólo se imprimen sólo los números de línea del programa.
Prueba, también, el siguiente comando:
POKE53265,11
Con él desactivarás la pantalla logrando que el programa se ejecute más rápido. Esto se debe a que la computadora no pierde tiempo en reescribir cada uno de los caracteres en pantalla. Utilizalo sólo cuando debes hacer las operaciones que demandan mucho tiempo. No olvides de usar el método de la inicialización. Para volver a activar la pantalla tipeą:
POKE53265,27
De todas maneras te sugerimos que mires la sección trucos del número 2, en donde hay un utilitario que permite que cualquier programa se autoejecute cuando se carga desde cinta o diskette

Continuamos con esta sección para que los lectores planteen sus consultas y sugerencias. Para eso deberán escribir a nuestra redacción: Cerrito 1320, (1010), Buenos Aires

Le agradecemos al Sr. Gustavo Ramos el habernos comentado los errores que se produjeron en el programa correspondiente a la nota de ASSEMBLER publicada en el número 1.
A continuación pasamos a describirlos:
En la línea 20 (fig. 3) debe decir IF en lugar de LF.
En la línea 49159 debe figurar 208 en lugar de 288.
En la línea 49166 sucede lo mismo que en la anterior.
En la línea 49187 debe ir 208 en lugar de 299.

TV

En primer lugar deseo felicitarlos por la estupenda revista que editan.
Poseo una C-64 y quisiera transformarla en PAL N, pero en algunos locales me aconsejaron que no deje abrir la máquina, que transforme el TV en otros negocios por 45, con 1 año de garantía. Ustedes qué me aconsejan?
Me gustaría intercambiar programas e ideas sobre la Commodore con otros chicos.
Muchas gracias.
María del Rosario Couste
Sarmiento 1099
(2183) Arequito - Santa Fe

Te sugerimos que modifiques el TV y no la computadora. Cada vez hay más negocios que se dedican a transformar el sistema NTSC a PAL N y viceversa.

Colaboraciones

Antes que nada deseo felicitarlos por la excelente revista que publican. Es única en su tipo.
Desearía saber como puedo (si puedo) colaborar con vuestra revista.
Les agradezco su atención.
Marcelo Julián Perez
Caseros - Bs. As.

Toda investigación, programa, etc., que hayas escrito, puedes enviarlos a nuestra redacción cita en Paraná 720 5º Piso - (1017) Cap. Fed.
En cierta forma ésto es una invitación a todos los lectores de Drean Commodore. Todo lo que ustedes crean interesante (programas, datos, trucos, etc.) envíenlos a la dirección antes mencionada (no olviden poner sus datos personales). El material debe ser inédito, es decir que nunca haya sido publicado en otras revistas .